Universidad
Carlos III de Madrid

Grado en Ingeniería en tecnologías de telecomunicación

Trabajo Fin de Grado:

Estudio y puesta en marcha de un sistema de streaming en Big Data

Autor: Andrea Isabel Estaire Muñoz
Tutor: Daniel Díaz Sánchez

Fecha: 21 de Junio de 2017

RESUMEN

En este proyecto se realiza un estudio de las tecnologías existentes de Big Data para poner en marcha un sistema basado en un flujo de datos continuo o streaming. El uso del Big Data es de gran utilidad en numerosos sectores de hoy en día, y en este proyecto aplicaremos esta tecnología en el desarrollo de un caso práctico y real.

Como partes fundamentales del proyecto tenemos, por un lado, el estudio de las tecnologías asociadas al Big Data y de las diferentes aplicaciones que las implementan (Twitter API, Kafka, Spark y MySQL), su instalación, configuración e integración, y por otro, el desarrollo de una aplicación usando estos sistemas para la resolución de un caso práctico: análisis de sentimiento a partir de los tweets de la red social Twitter.

En el desarrollo de la memoria, también se exponen trabajos relacionados, el impacto socio-económico que posee el proyecto, el marco regulador que engloba al Big Data, la planificación y el presupuesto necesario y, por último, las conclusiones obtenidas.

Palabras clave: Big Data, streaming, Twitter, tweets, análisis de sentimientos, Apache Spark Streaming, Apache Kafka, MySQL.

ABSTRACT

In this project, a study of the existing technologies of Big Data is realized to put on operation a system based on a continuous flow of data. Nowadays, the use of Big Data is very useful in many sectors, and in this project, we will apply this technology in the development of a practical and real case.

As fundamental parts of the project we have, on the one hand, the study of technologies associated with Big Data and the different applications that implement them (Twitter API, Kafka, Spark and MySQL), their installation, configuration and integration, and on the other hand, the development of an application using these systems focus on the resolution of a practical case: sentiment analysis of the tweets coming from the social network Twitter.

In the development of the report, related works are also presented, social and economic impact of the project, regulatory framework of Big Data, the planning and the necessary budget and, finally, the conclusions obtained.

Keywords: Big Data, streaming, Twitter, sentiment analysis, Apache Spark Streaming, Apache Kafka

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1 INTRODUCCIÓN

1.1 Contexto y motivación

La información siempre ha resultado ser una ventaja competitiva, para países, empresas, organizaciones, etc. Poseer esta ventaja es sumamente importante para poder lograr tus metas u objetivos y estar al nivel, o incluso por encima, de tus competidores.

La información que poseemos está creciendo exponencialmente en los últimos años, de tal manera, que ha supuesto un gran problema almacenarla, y por consiguiente poder analizarla e interpretarla.

Para poder solventar el problema de como almacenar dicho volumen de datos, las tecnologías han evolucionado de forma que han aparecido nuevos dispositivos de almacenamiento con mayor capacidad y de menor tamaño. Y respecto al análisis e interpretación de dicha información, nuevamente las tecnologías se han visto obligadas a aumentar la velocidad y capacidad del proceso para poder realizarlo, debido a que llegó un momento en el que la gran cantidad de información a procesar era tan grande que el tiempo necesario para analizarla se hizo demasiado lento, y provocó que cuando se obtenían los resultados, estos ya no eran válidos.

Es por ello, que aparece lo que se conoce como Big Data para poder analizar y obtener información valiosa de esta gran cantidad de datos que nos rodea hoy en día. [1]

En el presente proyecto se realizará una extracción de grandes cantidades de datos de una red social en tiempo real, haciendo uso de las tecnologías de Big Data, y se obtendrá una valoración de su reflejo sentimental a partir de las palabras de los tuits extraídos. Surge de la idea de que la gente hace uso de las redes sociales continuamente para expresar su opinión y sus sentimientos hacia cualquier tema. Por lo que extraer dicha información y analizarla puede ser muy útil, por ejemplo, para las empresas, ya que

pueden analizar cómo están reaccionando sus clientes hacia un determinado producto; y también puede tener gran utilidad en la política, debido a que les permite a las entidades políticas saber cuántas personas tienen en contra o a favor y hacia qué objetivos pueden dirigir sus campañas, entre otros muchos ejemplos.

1.2 Objetivos

Este proyecto ha perseguido los siguientes objetivos:

1. Realizar un estudio de Big Data y de las tecnologías y arquitecturas existentes para el análisis masivo de datos.

2. Realizar un análisis de sentimiento de la red social Twitter, para ello se ha de diseñar un sistema Big Data para extraer los tuits en streaming. Luego se han de tratar los datos extraídos para así obtener el sentimiento de dichos mensajes. Y por último almacenar y visualizar los resultados.

3. Evaluar el impacto socio-económico del proyecto.

4. Presentar el marco regulador asociado al Big Data.

5. Extraer conclusiones de dicho análisis y proponer mejoras.

1.3 Estructura del documento

1. **Introducción**: En este capítulo se describe el contexto en el que se desarrolla el proyecto y la motivación que lleva a la realización del mismo. Se establecen los principales objetivos y se expone la estructuración de la memoria.
2. **Estado del Arte:** En este capítulo se analiza el Big Data, los principales conceptos que le rodean, las fuentes de datos y algunas aplicaciones en las que su papel es de gran utilidad.
3. **Tecnologías para el análisis de Big Data:** En el capítulo tres se exponen las principales tecnologías que se utilizan para el análisis masivo de datos estáticos y datos en tiempo real (streaming). Algunas de ellas utilizadas en el desarrollo del caso realizado.
4. **Aplicación desarrollada:** En este capítulo se desarrolla la aplicación de análisis de sentimiento en Big Data. Se expone con más detalle lo que significa el

análisis sentimental en Twitter, se explica la arquitectura del sistema y la implementación que se ha llevado a cabo y la presentación gráfica de los resultados obtenidos.

5. **Trabajos realizados:** En este capítulo se exponen otros trabajos que se han realizado sobre el análisis de sentimiento.
6. **Impacto socio-económico:** En este capítulo se analiza el impacto social y económico del proyecto realizado.
7. **Marco regulador**: En el capítulo 7 se expone el marco regulador que rodea al Big Data.
8. **Planificación y presupuesto:** En este capítulo se recogen las fases de desarrollo del proyecto y el presupuesto necesario para llevarlo a cabo.
9. **Conclusiones y trabajos futuros:** En este capítulo se reflexiona sobre los objetivos que se han alcanzado en el proyecto y sobre futuras mejoras.
10. **Bibliografía:** Se enumeran las fuentes de información que se han utilizado en la elaboración del proyecto.
11. **Anexo A:** Se hace un resumen de la memoria en inglés.

CAPÍTULO 2

2 ESTADO DEL ARTE EN BIG DATA

2.1 Definición

Con el constante crecimiento de los datos en internet surge un nuevo término denominado Big Data. Big Data es el término que describe el proceso de recolección de un gran volumen de datos, que pueden estar tanto estructurados como no estructurados, y su posterior organización y análisis para encontrar la mayor cantidad de información posible, y obtener conocimiento de dichos datos.

Mientras que el concepto "Big Data" es relativamente nuevo, la recolección de datos para un análisis posterior existe desde hace muchos años, y un ejemplo de ello son las primeras bibliotecas que representan un primer intento de almacenar datos. Actualmente, todo lo que realizamos deja un rastro digital y junto con la expansión de internet y los avances en las tecnologías esta gran cantidad de datos almacenada crece exponencialmente día a día. En la siguiente figura podemos apreciar los antecedentes y la evolución de las tecnologías relacionadas con el Big Data [2]

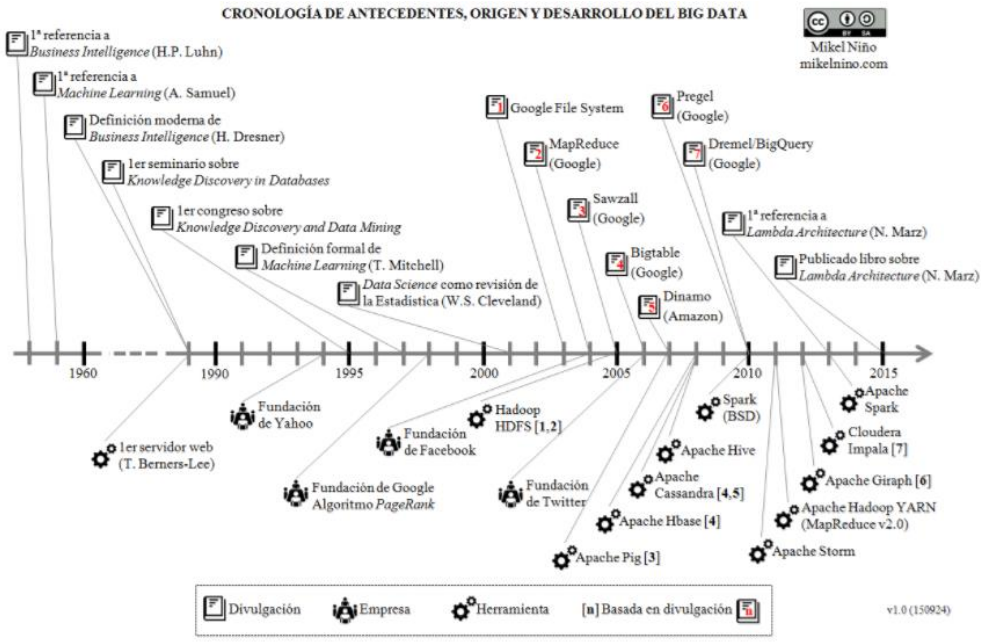

Figura 1: Cronología del Big Data [3]

2.2 Las 5 V's

Para caracterizar el sistema y explicar las ventajas de Big Data se utilizan cinco reglas o directrices, las 5 V's [4]:

- Volumen: Es la característica en la que primero pensamos cuando hablamos de Big Data ya que el volumen de datos con el que se trabaja es enorme. Según IBM, se generan 2.5 trillones de bytes en internet al día, es decir, este volumen aumenta a cada momento.
- Velocidad: Los datos fluyen a una velocidad sin precedentes, por lo que la velocidad en la que se analizan es sumamente importante. El análisis en tiempo real o streaming toma cada vez más valor entre las actividades de las empresas para poder tomar decisiones y para que dicho tratamiento suponga ventajas respecto a otras compañías.
- Variedad: Los datos poseen muchos tipos de estructuras y formatos. Es por ello, por lo que Big Data debe estar preparado para poder analizar todo tipo de datos: desde datos estructurados, numéricos, documentos de texto no estructurados, video, audio, etc.
- Veracidad: En ocasiones se hace inevitable dudar sobre el grado de veracidad de los datos, y es por ello, que se deben filtrar para asegurarnos del aprovechamiento y veracidad de la información extraída.
- Valor: Es el aspecto más relevante de Big Data, a medida que aumenta el volumen y complejidad de los datos, su valor marginal disminuye considerablemente, debido a su dificultad de explotación.

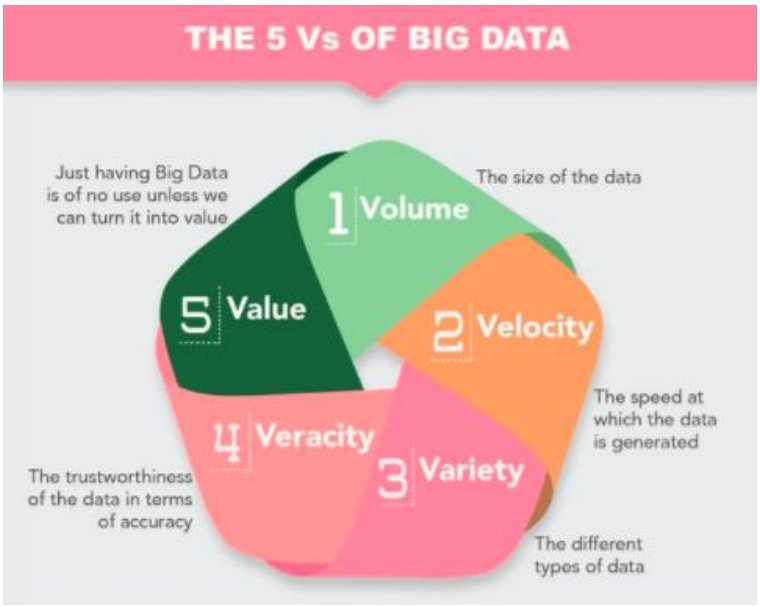

Figura 2: Las 5V´s del Big Data [5]

2.3 Fuentes de datos

Las fuentes de datos [6] desde donde podemos extraer los datos son muy variadas, esto provoca que no haya homogeneidad en la forma de extraerlos y posteriormente tratarlos. Existe un amplio abanico de fuentes desde donde podemos extraer información: open data, redes sociales, internet de las cosas (IOT), bases de datos de log, red en tiempo real, etc.

El objetivo del Big Data es aprovechar los datos de este tipo de fuentes de las que disponemos.

- Open Data: Se basa en la publicación en internet de datos de numerosas instituciones y comunidades que han decidido publicar para que se pueda sacar información relevante de ellos. Por ejemplo: encuestas, precios, consumos energéticos…
- Redes sociales: Día a día se genera gran cantidad de información útil en las redes sociales que puede ser aprovechada. Las dos redes sociales desde las que podemos extraer más información son Twitter y Facebook. Por ejemplo: Opiniones, sentimientos, imágenes, vídeos, etc.

- Sensores (IoT): Se puede obtener información relevante de sensores ya que permiten capturar las magnitudes físicas o químicas y convertirlas en datos, como por ejemplo una red de sensores de control de estrés informa sobre el estado estructural en edificios, puentes o carreteras.
- Red y base de datos de log. Analizando en tiempo real esta información podemos descubrir posibles intentos de ataques a nuestra red.

También cabe destacar que las fuentes de datos más comunes entre las empresas, según un reciente estudio de la empresa Cisco, no es otra que sus propios datos contenidos dentro de la compañía. [7]

2.4 Aplicaciones y usos del Big Data

Una vez explicado el Big Data vamos a ver que utilidades y soluciones aporta al mundo real el uso de esta tecnología. En los últimos años podemos encontrar ejemplos de uso del Big Data que han servido tanto para predecir tendencias como para generar nuevas oportunidades en acciones de marketing. Vamos a ver algunos de ejemplos [8] en los que la utilización del Big Data ha aportado grandes ventajas:

- Macy's: se trata de uno de los comercios más importantes de Estados Unidos. Utilizando la tecnología del Big Data ha conseguido una gran mejora en sus ingresos y en la experiencia del usuario. Ha conseguido reducir en 500.000 dólares el gasto anual, ya que gracias a la velocidad de análisis Macy´s puede saber con exactitud el impacto de sus noticias o la satisfacción y los gustos de sus clientes. Además, han conseguido que se reduzcan las bajas de suscripciones un 20%. Debido al control que posee gracias al uso de esta tecnología, Macy´s puede controlar sus 73 millones de artículos de venta prácticamente en tiempo real.
- Los equipos de la NBA**:** Utilizan la analítica de los datos a la hora de preparar su estrategia y tomar las mejores decisiones en un determinado partido. Con esto consiguen establecer las demandas de los fans y les facilita las cosas a la hora de establecer acciones de marketing, expandir el mercado, etc.
- La reelección de Obama: El expresidente de los Estados Unidos utilizó Big Data para su reelección en 2012. Realizaron numerosos análisis, y tras el primero la

campaña de Obama persiguió tres aspectos: recoger los datos de los votantes convencidos, dirigirse hacia los votantes dudosos de una manera más eficaz y asegurar el voto del electorado. Además, con su analítica, el equipo de Obama pudo optimizar la comunicación y no malgastó recursos, ni tiempo ni dinero en los votantes que no estaban a favor de su partido.

Estos son solo algunas aplicaciones del Big Data entre muchas otras que han obtenido beneficios haciendo uso de él.

En la figura 3, podemos ver todas las características que engloban al Big Data, como su infraestructura, sus análisis, sus aplicaciones y sus fuentes:

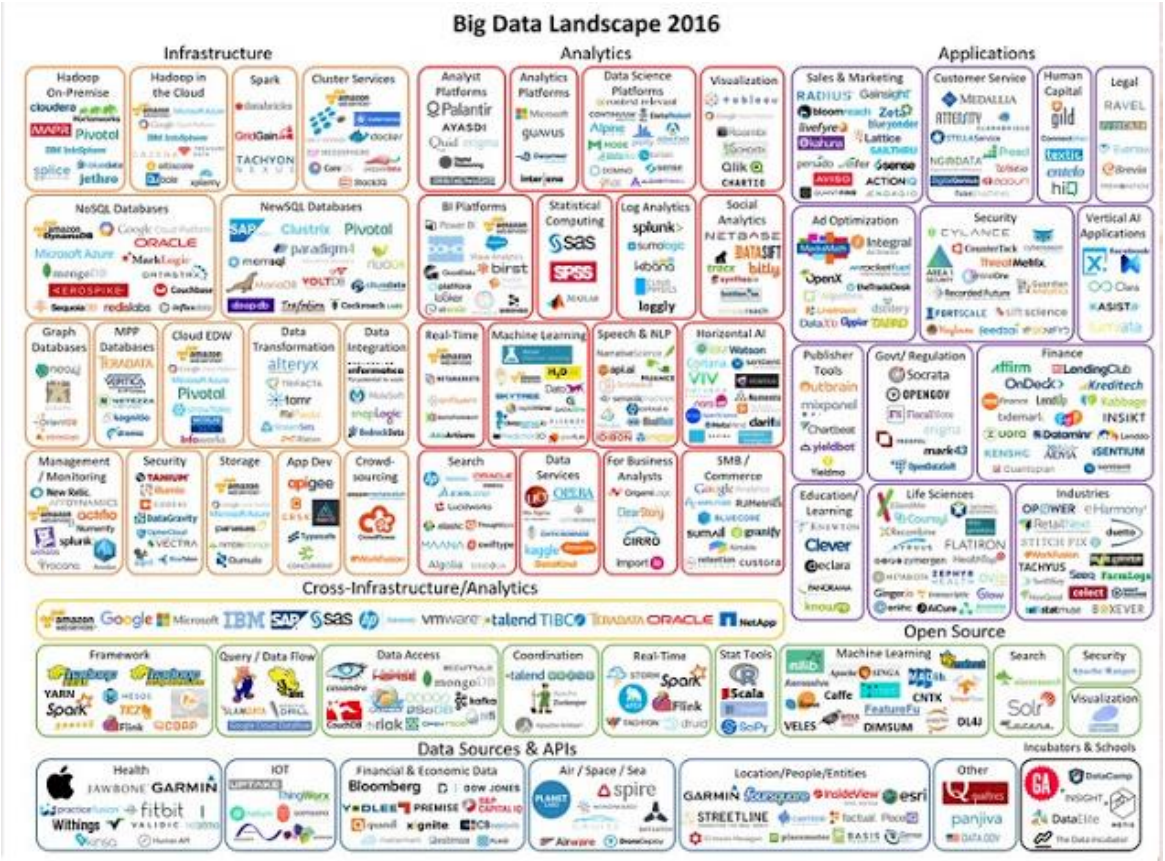

Figura 3: Panorámica del Big Data 2016 [9]

CAPÍTULO 3

3 TECNOLOGÍAS PARA EL ANÁLISIS DE BIG DATA

En este capítulo se realizará un análisis de las principales herramientas existentes para analizar datos en Big Data. Las diferenciaremos entre las utilizadas para análisis de datos estáticos y análisis de datos en tiempo real.

3.1 Sistemas Big Data

3.1.1 Apache Hadoop

Apache Hadoop es un sistema de código abierto que facilita el almacenamiento, procesamiento de datos y permite analizar grandes cantidades de información con rapidez y con gran tolerancia a fallos.

Hadoop aparece en 2004 cuando el ingeniero Doug Cutting redacta un documento sobre diferentes técnicas para manejar grandes volúmenes de datos, desgranando éstos en cantidades cada vez más pequeñas para poder analizarlos. El desarrollo de la plataforma se completa finalmente en el año 2008. [10]

Figura 4: Logo de Apache Hadoop

Entre sus ventajas [11] se encuentran:

- Permite realizar un procesamiento distribuido de grandes cantidades de datos en clústeres de manera sencilla escalando desde pocos ordenadores hasta miles de máquinas, es decir, está orientado hacia la computación distribuida.
- Sus dos atributos más significativos son la escalabilidad y la fiabilidad, debido que es capaz de recuperarse de manera automática de posibles fallos y de forma que no entorpezca al usuario.
- Es de código abierto y compatible con múltiples plataformas.

Los módulos que componen Hadoop [12] son:

- **Hadoop Common**

Contiene las bibliotecas y los sistemas de archivos necesarios para ejecutar Hadoop.

- **Hadoop YARN (Yet Another Resource Negotiator)**

YARN se trata de una plataforma de gestión y planificación de los recursos del clúster. Se basa en un gestor central y un gestor para cada nodo.

- **MapReduce**

Se trata del modelo de programación utilizado por Hadoop, para procesar grandes cantidades de datos en paralelo.

Se basa en la realización de dos operaciones, Map (mapear) y Reduce (reducir), a los datos almacenados en el clúster [13].

  - La función Map se encarga de realizar diferentes operaciones sobre el conjunto de datos que ha recibido, de forma que los reduzca en subconjuntos los cuales están formados por pares de clave y valor, o también llamadas tuplas. Se aplica en paralelo para cada dato de entrada, y toma uno de estos pares de datos en un dominio de datos, y devuelve una lista de pares en otro dominio diferente:

Map(k1,v1) -> list(k2,v2)

  - La función Reduce que también es aplicada en paralelo, recibe esas tuplas y realiza operaciones adicionales para convertirlas en un subconjunto de datos aún más pequeño, combinando valores con la misma clave en un mismo resultado, produciendo una colección de valores para cada dominio:

Reduce(k2, list (v2)) -> list(v3)

Por lo tanto, después de realizar ambas operaciones vemos transformada una lista de pares clave-valor en una lista de valores.

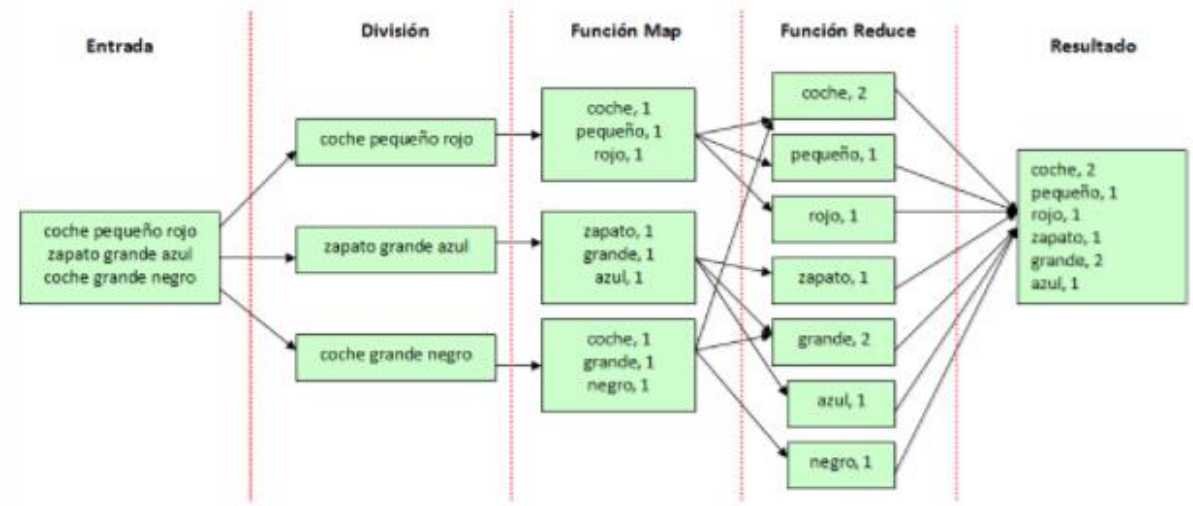

Figura 5: Funcionamiento Map-Reduce [14]

El módulo MapReduce de Hadoop está compuesto por [12]:

- JobTracker: Hay uno por clúster. Se encarga de distribuir los trabajos de MapReduce al resto de TaskTrackers para que las realicen. Si un TaskTracker falla el trabajo se replanifica.
- TaskTracker: Hay uno por nodo del clúster. Realiza las operaciones de MapReduce al conjunto de datos que recibe. Posteriormente devuelve los resultados o los almacena.

- **Hadoop Distributed File System (HDFS)**

HDFS [15] es el sistema de archivos distribuido de Hadoop que está inspirado en Google File System (GFs). Posee gran escalabilidad, permite distribuir la información en clústeres y se encuentra escrito en Java.

HDFS se trata de un conjunto de nodos distribuidos en los que se almacena información y sus réplicas. Los datos se dividen en diferentes bloques y son almacenados en diferentes máquinas o nodos para garantizar una buena tolerancia a fallos y mayor fiabilidad del sistema, de forma que si se pierde la comunicación con uno de ellos la información no se pierda. Esta manera de distribuir la información implica un aumento del rendimiento porque al encontrarse los datos en varios sitios se pueden procesar de una manera más rápida.

Los nodos se agrupan en clústeres, y en cada clúster encontramos un NameNode y uno o varios DataNodes:

- NameNode: Es el nodo primario. Se ocupa de llevar un control de acceso y contiene un registro de la distribución de todos los datos almacenados en el sistema. Coordina las operaciones de apertura de archivos, creación o modificación de los mismos. Determina la asignación de los bloques de información a los diferentes DataNodes.
- DataNode: Los DataNodes contienen la información y se encargan de ejecutar las peticiones de lectura y escritura del cliente.

En la figura 6, podemos ver una representación de la arquitectura de HDFS:

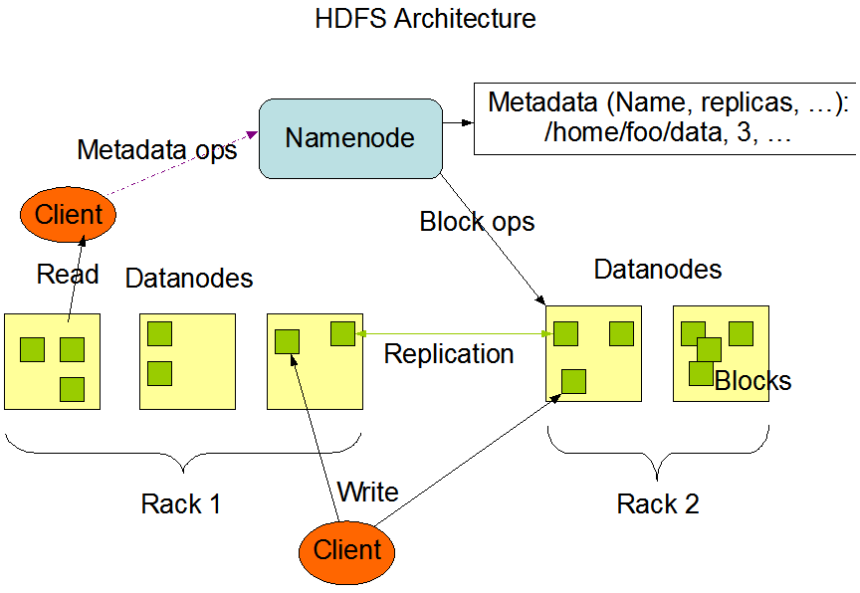

Figura 6: Arquitectura HDFS [15]

3.1.2 Apache Spark

Spark [16] es un framework de código abierto para analizar grandes volúmenes de datos, construido sobre Hadoop. Fue diseñado para cumplir tres prioridades: velocidad, facilidad de uso y capacidad avanzada de analítica.

Apache Spark es la nueva estrella del Big Data. La plataforma está escrita en el lenguaje scala, pero puede operarse en otros tres lenguajes de programación: Java, R y Python.

Posee una velocidad de cálculo en memoria 100 veces más rápida y de cálculo en disco 10 veces más ágil que la que ofrece Hadoop MapReduce.

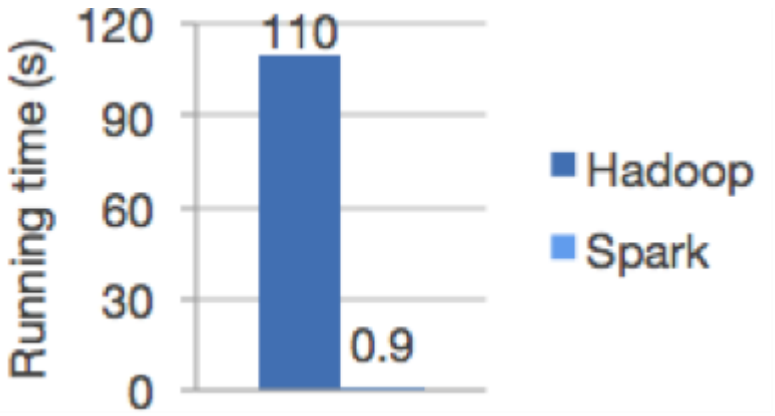

Figura 7: Comparación de velocidad en Spark y Hadoop [16]

La clave de su rendimiento es que Spark proporciona la ventaja de que los datos pueden ser cargados en una memoria de clúster y ser consultados y procesados repetidamente, esto hace que Spark sea un buen candidato en el que poder implementar algoritmos de "Machine Learning".

Podemos ejecutar la plataforma en Hadoop, en EC2, en Apache Mesos, en modo clúster o en la nube y, además, desde Spark se puede acceder a muchas bases de datos como MySQL, Cassandra o HBase, entre otras.

Spark incorpora herramientas de gran utilidad, como son:

- La librería MLib: para poder implementar soluciones de aprendizaje automático.
- GraphX: la Api para la computación con grafos.
- Spark Streaming: para el procesamiento de datos de entrada constante en tiempo real de millones de datos en clústeres, de la que hablaré con más detalle en el siguiente punto.
- Spark SQL: para la explotación de los datos por medio de lenguaje SQL.

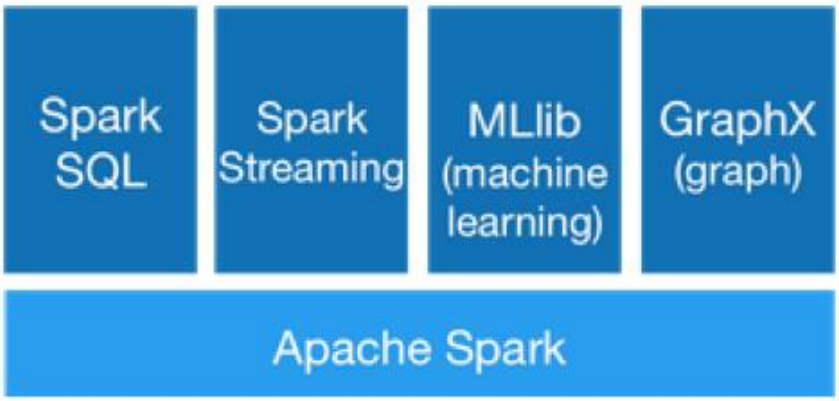

Figura 8: Herramientas de Apache Spark [16]

Resilient Distributed Datasets (RDD)

Resilient Distributed Datasets (RDD) [17]es la unidad de datos fundamental que el motor de procesamiento de Spark consume. [18] Representa una colección inmutable de objetos, tolerante a fallos que pueden ser operados en paralelo. Los RDD son capaces de contener cualquier tipo de objetos Python, Scala o Java, incluidas las clases definidas por el usuario.

Las operaciones básicas de los RDD son map, filter y persist. Cada RDD, internamente, se compone de cinco características principales:

- Una lista de particiones
- Una función para calcular cada división

- Una lista de dependencias con otros RDDs
- Opcionalmente, una partición para RDD de valor—clave.
- Opcionalmente, una lista de ubicaciones preferidas donde calcular cada división (por ejemplo, ubicaciones en bloque para un archivo HDFS)

Toda la programación y ejecución en Spark se realiza basándose en estos métodos, permitiendo que cada RDD implemente su propia forma de procesamiento.

SparkContext

SparkContext [19] es el punto de entrada principal para la ejecución de Spark. Representa la conexión a un clúster Spark y puede utilizarse para crear RDD, acumuladores y variables de difusión en ese clúster.

Una vez se instancia SparkContext, se obtienen ejecutores en los nodos de cada clúster. Dichos ejecutores son procesos que se encuentran dentro de cada nodo, y sirven para almacenar datos y ejecutar cálculos. El SparkContext envía a cada ejecutor el código para la aplicación y las tareas que el ejecutor tiene que realizar.

3.2 Sistemas de streaming

3.2.1 Apache Spark Streaming

Apache Spark Streaming [20] es una herramienta de Apache Spark utilizada para el procesado de datos de flujo constante. Puede recibir datos de numerosas fuentes como, por ejemplo, Twitter, Facebook, Apache Kafka o Apache Flume, así como también es capaz de recibir datos recogidos de sensores o dispositivos conectados a través de sockets TCP. También es capaz de procesar datos almacenados en sistemas de archivos como HDFS o Amazon S3.

Internamente, Spark Streaming recibe un flujo de datos y los agrupa en lotes de datos, que pasan a ser procesados por Spark, que a su vez devuelve lotes de datos ya procesados.

Podemos ver representado este funcionamiento en la figura 9:

Figura 9: Spark streaming [20]

En líneas generales, Spark Streaming recibe un flujo de datos continuo y lo convierte en un flujo discreto, llamado DStream, que se encuentra formado por paquetes de datos. Un DStream puede ser creado al recibir datos de distintas fuentes o aplicando operaciones sobre otros DStream. A su vez, un DStream es representado como una secuencia de objetos RDD (Resilient Distributed Data).

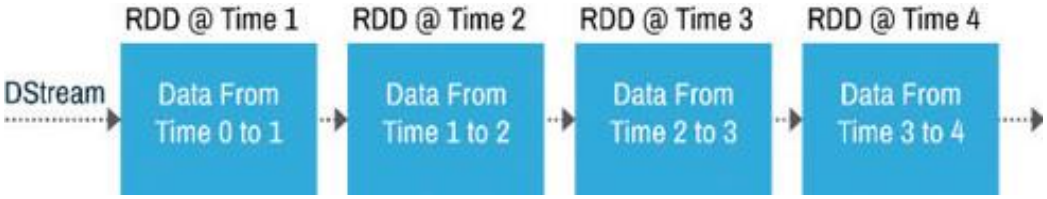

Figura 10: Composición de un DStream [20]

Cualquier operación que es aplicada a un DStream se le aplica a cada RDD que contiene en su interior. Podemos verlo en la figura 11, en donde se pretende realizar la operación de contar las palabras de las diferentes líneas de un determinado texto contenidas en los RDDs de un DStream, y esta operación se realiza a cada uno de los RDD.

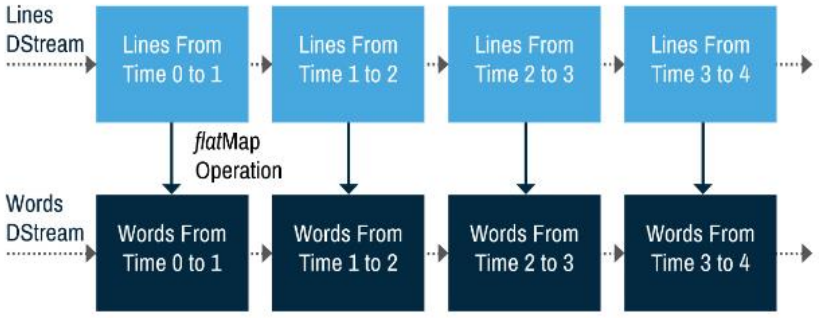

Figura 11: Operaciones sobre RDDs de un DStream [20]

StreamingContext

Para las aplicaciones basadas en flujos de datos, es utilizado StreamingContext en vez de SparkContext, ya que StreamingContext soporta DStreams.

StreamingContext [21] es el punto de entrada principal para la ejecución de Spark Streaming. Proporciona la información básica, como la URL del clúster y el nombre del trabajo a realizar, y también contiene métodos que son utilizados para crear DStream.

Micro-lotes

Los micro-lotes [20] (micro-batches) sobre los que opera Spark Streaming tienen un tiempo de intervalo entre unos y otros. Esta forma de operar con flujos de datos continuos puede tener algunas consecuencias.

La principal consecuencia es que los datos puede que se reciban y procesen en un orden diferente al que realmente sucedieron.

Por un lado, para que esto no ocurra, se puede reducir el tiempo del intervalo a menos de un segundo, lo que nos proporcionaría un análisis prácticamente en tiempo real, pero necesitaríamos invertir un alto costo en recursos para el procesamiento de estos datos. Por lo que no siempre es ésta la mejor opción.

Es por ello, por lo que son utilizadas plataformas como Apache Kafka, entre otras, que explicaremos en el siguiente apartado, para garantizar que los datos lleguen en el orden exacto.

3.2.2 Apache Storm

Apache Storm [22] es un sistema de computación en tiempo real distribuido y de código abierto. Storm permite procesar flujos de datos de manera fiable y sin límites. Storm es simple y se puede utilizar con varios lenguajes de programación como Java, Python, Scala, C#, entre otros. Se puede integrar con bases de datos y con las tecnologías de colas.

A diferencia de Hadoop, que procesa los datos por lotes, Storm lo hace en tiempo real. Storm transforma los datos y los analiza dentro de un proceso continuo de entrada constante de información.

Apache Storm puede ser aplicado a números casos de uso, como análisis en tiempo real, aprendizaje en línea, y muchos otros. Entre sus fortalezas se encuentran que es escalable, tiene alta tolerabilidad a fallos y garantiza que todos los datos van a ser procesados.

Sigue una arquitectura [23] Maestro-Esclavo (Master-Slave) donde ninguna instancia mantiene estado y es gestionado por Zookeeper. El Maestro es denominado Nimbus y los esclavos se llaman Supervisor.

- Maestro o Nimbus: Es el responsable de realizar a asignación y monitorización de las distintas tareas en los nodos del clúster.
- Esclavo o Supervisor: Recoge y realiza las tareas que le son asignadas. Si uno de ellos falla, Zookeeper es el encargado de avisar al maestro para que redirija las tareas a otro esclavo.

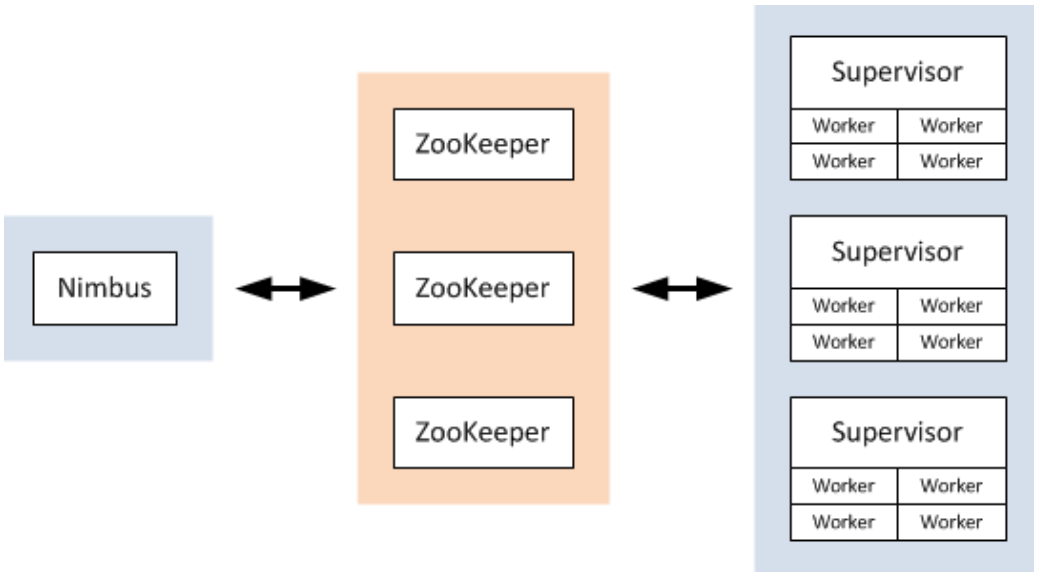

Figura 12: Arquitectura de Apache Storm [24]

Apache Storm [23] está formado por Topologies, que son árboles complejos donde se realiza el procesamiento y que a su vez están formadas por spouts y bolts.

- Spout: Encargado de recoger el flujo de datos de entrada.
- Bolt: Contiene las operaciones para realizar los cálculos.

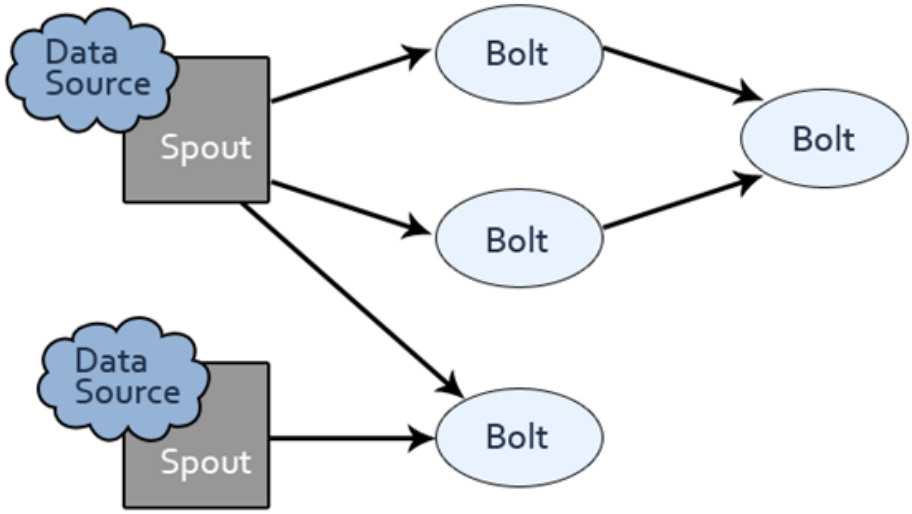

Figura 13: Topology Storm [25]

3.3 Sistema complementario para análisis de streaming

Existen una serie de tecnologías de colas que facilitan el análisis de datos en tiempo real, como pueden ser Apache Kafka o Apache Flume, entre otras.

Se explicará con más detalle Kafka [26] ya que se hace uso de ella en el caso implementado.

3.3.1 Apache Kafka

Apache Kafka [27] es un sistema de mensajería distribuido y una cola de gran capacidad con la que se puede manejar un gran volumen de datos y permite pasar mensajes de un punto final a otro. Kafka se integra muy bien con Apache Spark y Apache Storm para el análisis de datos en tiempo real. Los mensajes que llegan a Kafka se mantienen en el disco y son replicados dentro del clúster para hacer el sistema tolerante a fallos ante la pérdida de datos en algún nodo del clúster. Para poner en funcionamiento Kafka es necesario construirlo sobre el servicio de sincronización Zookeeper que es el encargado de mantener el estado en el clúster.

Aporta numerosos beneficios, entre los cuales se encuentran:

- Fiabilidad: Kafka es tolerante a fallos, debido a que sus datos se distribuyen, se particionan y se replican.
- Escalabilidad: Es escalable, sin llegar a tener un tiempo de inactividad.
- Durabilidad: Los mensajes persisten en el disco todo el tiempo que pueda llegar a ser necesario para asegurar el consumo de éstos.
- Rendimiento: Es muy rápido tanto en la publicación como en la suscripción de mensajes. Incluso, aunque haya Terabytes de datos almacenados, mantiene un buen rendimiento.

Kafka soporta la entrega de mensajes con baja latencia y garantiza la tolerancia a fallos en presencia de posibles fallos de la máquina. Tiene la capacidad de manejar un gran número de diferentes consumidores y de realizar 2 millones de escrituras por segundo.

Los componentes de Kafka son:

- Topics (temas): A cada flujo de mensajes que pertenecen a una misma categoría se le denomina Topic. Los datos son almacenados en Topics.
- Particiones: Los Topics se dividen en particiones. Una partición es un conjunto de segmentos de datos de igual tamaño.
- Réplicas: Son copias de seguridad de una partición para así evitar la posible pérdida de datos.
- Brockers (corredores): Son los responsables de mantener los datos duplicados. Cada Brocker puede contener cero o alguna partición por Topic.
- Clúster: Se puede expandir un clúster de Kafka sin llegar a tener ningún tiempo de inactividad. Estos clústeres son utilizados para administrar la persistencia y la replicación de los datos del mensaje.
- Productores: Editan los mensajes en uno o más Topics. Envían los datos a los Brockers de Kafka.
- Consumidores: Leen los datos de los Brockers. Los consumidores se subscriben a uno o más Topics y consumen esos mensajes.
- Líder: Es el nodo que se encarga de todas las lecturas y escrituras de una partición dada. Cada partición tiene un servidor que actúa como líder.
- Seguidor: Encargado de seguir las instrucciones que le encomiende su líder. En caso de que el líder falle, uno de los seguidores automáticamente debe pasar a serlo

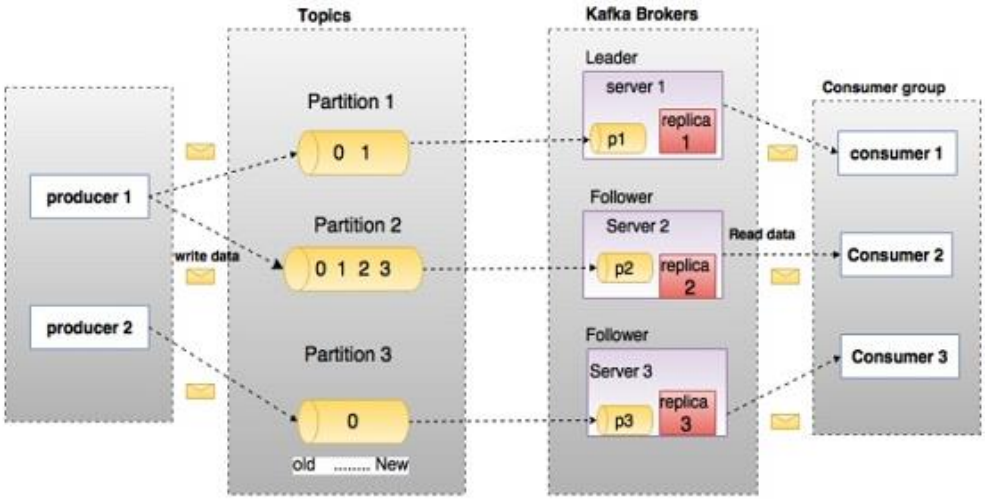

Figura 14: Componentes de Apache Kafka [27]

Los componentes de un clúster de Kafka son:

- Brocker (corredor): En el clúster existen múltiples Brockers, cuya función es mantener la carga balanceada. Utilizan Zookeeper para informar de su estado en el clúster. La elección del líder del Brocker también puede ser gestionada por Zookeeper.
- Zookeeper: Es utilizado para gestionar y coordinar los Brockers de Kafka. Si aparece un nuevo Brocker en el sistema como si un Brocker falla, Zookeeper se encarga de notificárselo al productor y al consumidor. Y así el productor y el consumidor sabrán cómo continuar coordinando sus tareas.

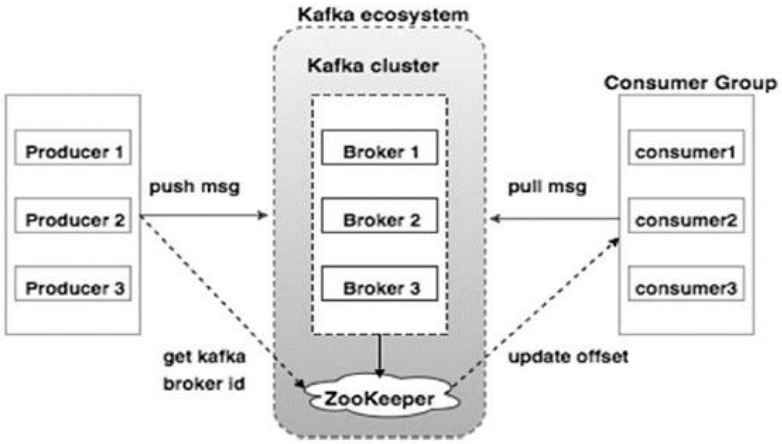

Figura 15: Clúster de Kafka [27]

Zookeeper

Zookeeper se trata de un servicio distribuido. Se utiliza para coordinar los consumidores y los Brockers. Si fracasa el líder de un Brocker, la elección del nuevo líder también la realizará Zookeeper.

CAPÍTULO 4

4 APLICACIÓN DESARROLLADA

4.1 Introducción

El trabajo realizado consiste en hacer un análisis de sentimiento de una red social. La red social seleccionada ha sido Twitter. Este análisis requiere del conocimiento y uso del Big Data debido a que es necesario procesar una gran cantidad de información en tiempo real.

Para llevarlo a cabo es necesario realizar la instalación, configuración y puesta en marcha de un sistema formado por una fuente de streaming (Twitter), un sistema para garantizar la integridad de la información recibida, es decir, para que no se pierda ningún dato recibido (Kafka), un sistema de análisis de la información en streaming (Spark Streaming) y una base de datos para el almacenamiento de los resultados del análisis (MySQL).

He decidido utilizar Spark Streaming ya que resulta fácil de usar y tiene un rendimiento muy bueno a la hora de analizar flujos de datos constantes. El lenguaje utilizado ha sido Java y como entorno de desarrollo se ha empleado Eclipse, ya que proporciona una buena interfaz de desarrollo desde donde poder realizar todo el ciclo de desarrollo de una aplicación.

Figura 16: Arquitectura de la aplicación desarrollada

4.2 Análisis de sentimiento de Twitter

El análisis de sentimiento [28] de mensajes en las redes sociales es también conocido como minería de opinión, y se trata del proceso a través del cual se puede determinar el tono emocional que se encuentra detrás de un texto, clasificando la polaridad expresada en positiva, negativa o neutra.

Dada la gran cantidad de emociones, opiniones y valoraciones que se ven diariamente en las redes sociales, resulta de gran utilidad extraer dicha información y poder averiguar si el sentimiento predominante es positivo o negativo. El objetivo primordial es extraer los términos que expresen un sentimiento en particular para conocer la opinión y el estado sentimental de los usuarios, para así poder determinar su impacto o poder anticiparse a una determinada reacción.

Sin embargo, nos encontramos con algunas dificultades debido a que el lenguaje es muy complejo y en ocasiones ambiguo, por lo que analizar a la perfección todos los matices y expresiones coloquiales resulta prácticamente imposible. Y sobre todo cuando el análisis se realiza de los mensajes extraídos de las redes sociales, ya que es un lenguaje muy coloquial y en numerosas ocasiones se cometen faltas de ortografía.

En lugar de analizar todas las palabras se ha selecciona una lista de palabras que expresan emociones con la que se han comparado dichos mensajes y extraído sus conclusiones.

4.2.1 Selección de la lista de palabras

La lista de palabras seleccionada ha sido AFINN [29]. Proviene de una lista anterior denominada ANEW, propuesta por Bradley y Lang en 1999, provee puntuaciones de sentimientos de 1.000 palabras inglesas. Estos rankings fueron calculados de acuerdo con tres reacciones psicológicas de una persona hacia una determinada palabra: el nivel de placer, el grado de control y la intensidad de la emoción. La reacción del nivel de placer, que puntúa la palabra según placentera o no, es el valor más útil para el cálculo de la polaridad de las palabras.

AFINN creada manualmente por Finn Årup Nielsen en 2011, está más centrada en el lenguaje usado en las plataformas de internet. ANEW se hizo antes del despliegue de estas plataformas, y es por esto por lo que muchas palabras que son usadas habitualmente en las redes sociales no están incluidas. Era necesaria una actualización de ANEW, por lo que la lista AFINN se adapta más al lenguaje de hoy en día e incluye palabras coloquiales y obscenas que aparecen normalmente en las redes sociales.

Se compone de 2477 palabras inglesas que expresan sentimientos y están calificadas entre menos cinco para sentimientos negativos, y más cinco para sentimientos positivos; siendo el valor cero asignado a sentimientos neutros.

Las palabras respecto a su calificación se encuentran separadas por tabuladores y por líneas.

Se ha elegido utilizar esta lista ya que la mayoría de la población en el mundo habla y escribe en lengua inglesa y, por otro lado, esta lista ha sido evaluada y utilizada en numerosos proyectos [30].

4.3 Arquitectura del sistema

4.3.1 Fuente de datos en streaming

4.3.1.1 Twitter

La fuente de datos de que la que se va a llevar a cabo el análisis es la red social Twitter debido a que en la actualidad millones de tuits son publicados cada día.

Twitter [31] se creó en marzo de 2006 por Jack Dorsey, Evan Williams, Biz Stone y Noah Glass y se lanzó en julio. El 15 de enero de 2009, un avión de US Airways se estrelló en el río Hudson en Nueva York. Y fue una foto en Twitter la que dio a conocer la noticia, incluso antes que los medios de comunicación, lo que marcó la importancia de Twitter en las noticias.

En 2016 [32] se recogieron los siguientes datos estadísticos: existen 310 millones de usuarios activos al mes, han sido creadas un total de 1,3 mil millones de cuentas y se publican 500 millones de tuits al día.

Es por ello, que a través de esta red social podemos detectar los temas más populares y conocer qué está ocurriendo en un preciso momento en cualquier lugar del mundo. Lo que empezó siendo una plataforma inicialmente utilizada por jóvenes, Twitter se ha convertido hoy en día en una de las fuentes de información más poderosas.

Twitter es una manera rápida y sencilla de compartir información. El tamaño máximo de un tuit es de 140 caracteres. Por lo general, en Twitter encontramos mensajes subjetivos y numerosas opiniones de los usuarios hacia los temas del momento, y es por esto por lo que realizar un análisis sentimental de Twitter es una idea que resulta bastante interesante.

La extracción de datos de Twitter es sencilla, debido a que la información es pública y hay bastantes herramientas que simplifican la extracción de estos tweets para poder realizar un análisis.

Twitter pone a disposición de los desarrolladores su API [33] para la extracción de datos. Para poder extraer datos de la API debemos entrar en la aplicación y generar un token haciendo clic en el botón "Create my access token"; y un token de acceso y un token de acceso secreto serán creados para nuestra cuenta y nuestra aplicación.

Figura 17: Access token de Twitter [33]

Una vez creado, si entramos de nuevo en la aplicación de Twitter y pulsamos sobre la aplicación que se ha creado en la pestaña de "Keys and Access Tokens" se encuentran los datos que son necesarios para poder extraer los tuits de Twitter:

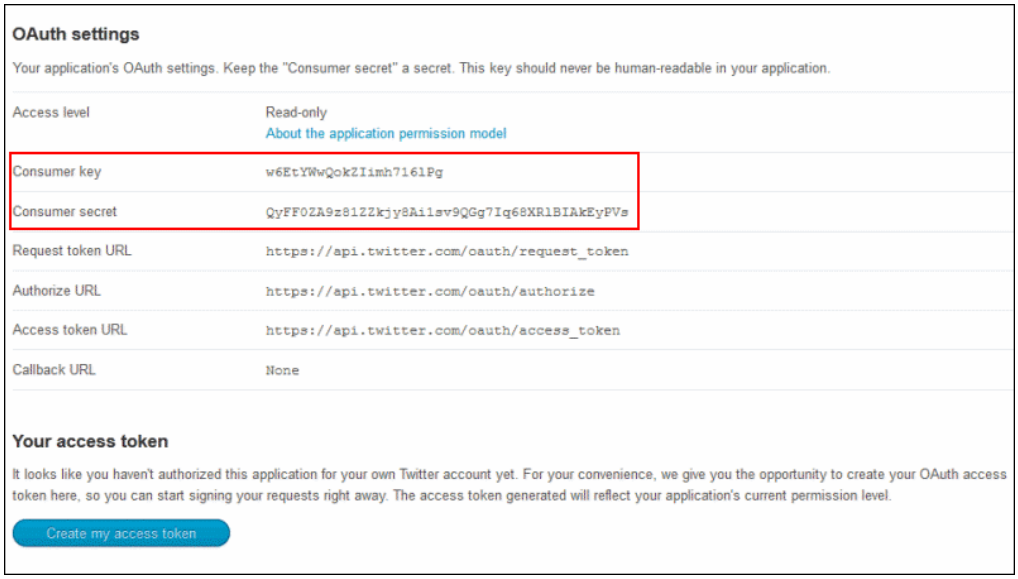

Figura 18: Consumer key y Consumer secret de Twitter [34]

4.3.2 Recogida y tratamiento de datos

En esta fase del proyecto han sido elegidas las plataformas: Apache Kafka, para la monitorización y recogida de los datos de Twitter, y Apache Spark Streaming [35] , uno de los módulos de Apache Spark, para el posterior tratamiento del flujo de datos y su análisis. Ambas se integran muy bien para el análisis de datos en tiempo real.

Apache Kafka es muy rápido en lecturas y escrituras, por lo que lo convierten en una herramienta muy buena para comunicar flujos de datos que son generados a gran velocidad con Spark Streaming. La combinación de ambas asegura que la aplicación sea robusta.

Los funcionamientos de ambos sistemas han sido explicados previamente.

4.3.3 Almacenamiento de los resultados

El almacenamiento de los resultados obtenidos al analizar los tuits se ha llevado a cabo en la base de datos MySQL [36].

4.3.3.1 MySQL

Se trata de una base de datos [37] relacional de código abierto desarrollada por Oracle Corporation, la cual es considerada la más popular del mundo. MySQL fue inicialmente desarrollado por David Axmark, Allan Larsson y Michael Widenius. En 2010, fue comprada por Oracle Corporation.

MySQL es patrocinada por una empresa del sector privado, la cual posee el copyright de casi todo el código, al contrario de otros proyectos donde el software se desarrolla por la comunidad y el código es abierto.

En su mayor parte, se encuentra desarrollado en ANSI C Y C++. MySQL es utilizado por numerosos sitios web grandes y populares como, por ejemplo, Facebook, Twitter, Wikipedia, Youtube, etc.

4.4 Diseño e implementación del sistema

4.4.1 Introducción

El sistema implementado se ha desarrollado sobre una máquina de Windows 10, utilizando Eclipse [38] y el lenguaje de programación Java versión 1.8.

El sistema está formado por tres aplicaciones: KafkaTweetProducer, SparkTweetConsumer y VisualizacionGrafica.

Cada una de estas aplicaciones de consola se ejecutan por separado, pero KafkaTweetProducer y SparkTweetConsumer deben ejecutarse simultáneamente ya que cada una realiza un papel complementario. KafkaTweetProducer accede a los tuits y los deja en Kafka a disposición de Spark representado por la aplicación SparkTweetConsumer, que los lee, los procesa y el resultado lo deposita en la base de datos de MySQL para que sean accedidos y presentados gráficamente por la aplicación VisualizacionGrafica.

Como ya se ha comentado anteriormente, el objetivo de la aplicación a desarrollar es analizar el sentimiento global de la red social Twitter mediante la identificación y peso de determinadas palabras dadas por el diccionario denominado AFINN dentro de un determinado contexto, por ejemplo, eligiendo todos los tuits en los que aparece la palabra "Trump". Este análisis se ha hecho en base a coger lotes de tuits en una ventana temporal de 100 segundos, durante 900 segundos que equivalen a 15 minutos, es decir, se han analizado 9 lotes en un momento en el que no había acontecimientos especiales. También la API de Twitter permite filtrar (seleccionar) los tuits que se desean analizar por una determinada palabra. El Análisis se ha realizado utilizando como filtro la palabra "Trump". Se ha elegido este filtro por ser un personaje muy relevante y de actualidad.

A continuación, se describe cada una de estas aplicaciones.

4.4.2 KafkaTwitterProducer

Esta aplicación se encarga de conectarse a Twitter para leer todos los tuits y almacenarlos en las colas persistentes de Kafka. Posteriormente la aplicación SparkTweetConsumer leerá estos tuits y realizará el análisis.

El diseño de clases implementado y la arquitectura de la aplicación es la siguiente:

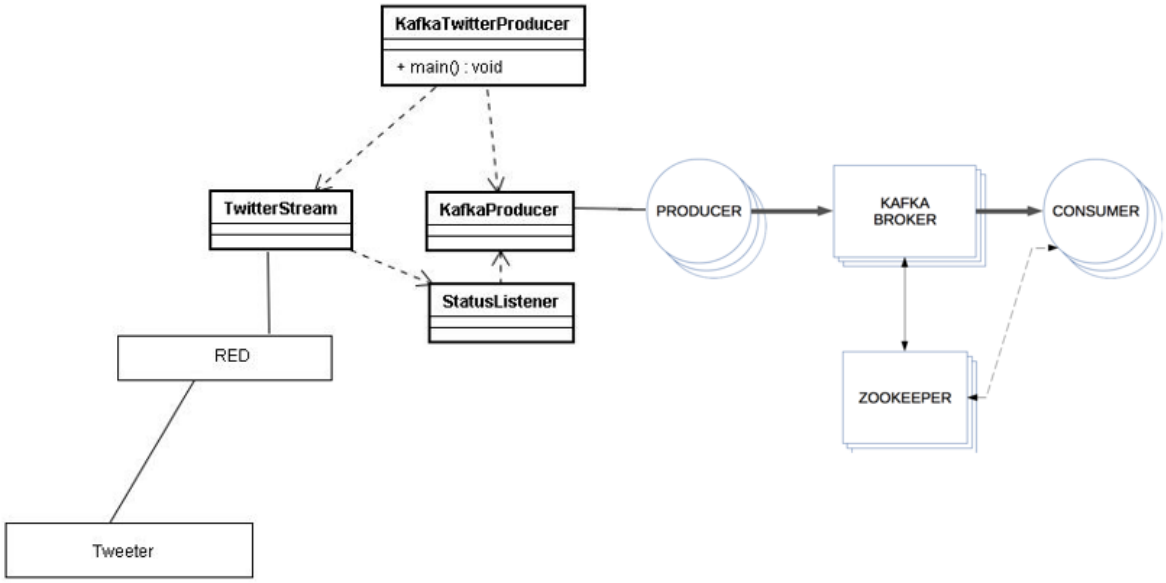

Figura 19: Diseño de clases KafkaTwitterProducer

En la figura podemos ver un componente que se denomina Zookeeper, es un servidor en el que se apoya el servidor de Kafka, y sirve para mantener el estado de la información distribuida, es decir, que todas las máquinas tengan la misma información y en el mismo orden en las colas de los diferentes tópicos.

Estos dos servidores hay que lanzarlos desde un terminal Windows antes de ejecutar la aplicación.

Puesta en marcha de Zookeeper y el servidor de Kafka:

```
C:\Users\Andrea\Desktop\PROYECTOTFG\spark\kafka_2.11-0.10.2.0\bin\windows>start
zookeeper-server-start.bat ..\..\config\zookeeper.properties

C:\Users\Andrea\Desktop\PROYECTOTFG\spark\kafka_2.11-0.10.2.0\bin\windows>start
kafka-server-start.bat ..\..\config\server.properties
```

Figura 20: Puesta en marcha de Zookeeper y el servidor de Kafka

Las clases más relevantes son las siguientes:

- **KafkaTwitterProducer**

Esta es la clase principal de la aplicación. En esta clase utilizamos clases de la librería twitter4j [39]. Con twitter4j se puede integrar fácilmente una aplicación Java con el servicio de Twitter.

En esta clase creamos el objeto productor de Kafka, que será el que escriba un tópico y un mensaje (tuit) en la cola de Kafka. Creamos el flujo twitterStream que será el encargado de leer los tuits e invocar al método onStatus del listner cuando reciba un tweet.

```
Producer<String, String> producer = new KafkaProducer<String,
String>(props);
TwitterStream twitterStream = new
TwitterStreamFactory().getInstance();
StatusListener listener = new StatusListener() {

public void onStatus(Status status) {

   Status ret=status;
   System.out.println("Tweet: " + ret.getText());
   producer.send(new ProducerRecord<String, String>(
     topicName,
     ret.getText()
   )
);
```

A continuación, definimos los filtros de idioma de los tuit y palabras clave.

```
twitterStream.addListener(listener);
FilterQuery query = new FilterQuery().track(keyWords);
twitterStream.filter(query);
query.language(new String[]{"en"});
```

- **Twitter4j.TwitterStream**

Es la clase encargada de leer los tuits de Twitter y a través de la clase twitter4j.StatusListner ponerlos a disposición de la clase KafkaProducer.

- **KafkaProducer**

Esta es la clase que interactúa con Kafka escribiendo los tuits en el sistema Kafka y asociándolos a un tópico.

- **Maven**

La configuración del proyecto de Eclipse utiliza Maven para el acceso a los paquetes necesarios en la generación del proyecto.

```
<dependencies>
		<dependency>
		    <groupId>org.apache.kafka</groupId>
		    <artifactId>kafka_2.12</artifactId>
		    <version>0.10.2.0</version>
		</dependency>
		 <dependency>
		    <groupId>org.twitter4j</groupId>
		    <artifactId>twitter4j-core</artifactId>
		    <version>4.0.6</version>
		</dependency>
		<dependency>
		      <groupId>org.twitter4j</groupId>
		      <artifactId>twitter4j-async</artifactId>
		      <version>4.0.6</version>
		      <scope>test</scope>
		</dependency>

		<dependency>
		    <groupId>org.twitter4j</groupId>
		    <artifactId>twitter4j-stream</artifactId>
		    <version>4.0.6</version>
		</dependency>

</dependencies>
```

4.4.3 SparkTweetConsumer

Esta aplicación es el corazón del sistema y se encarga de leer los tuits de Kafka y realizar el proceso. El resultado del proceso se almacena en una tabla de MySQL para su posterior análisis. Esta información posteriormente es procesada por la aplicación VisualizacionGrafica para tener una imagen visual del resultado del proceso realizado.

El diseño de clases implementado y la arquitectura de la aplicación es el siguiente:

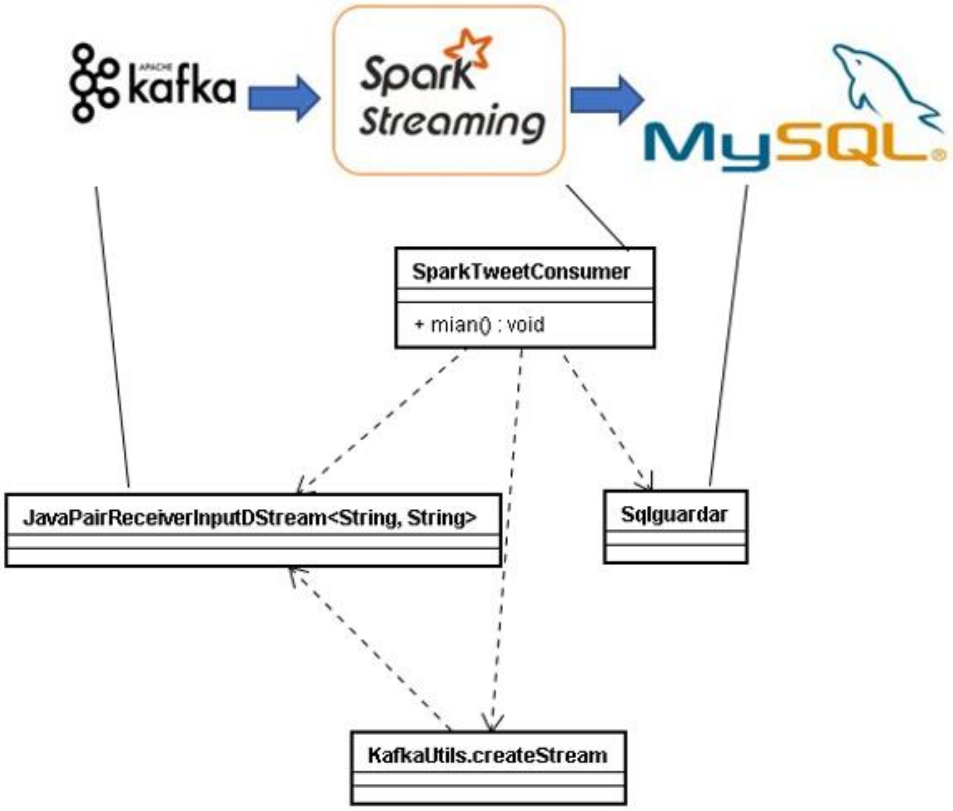

Figura 21: Diseño de clases de SparkTweetConsumer

Las clases son las siguientes:

- **SparkTweetConsumer:**

Esta es la clase principal. Se hace uso del método KafkaUtils.createStream para obtener un flujo de recepción suscrito a Kafka. A partir de la recepción de los tuits, es cuando se realiza el proceso de análisis sentimental, consistente en contar las palabras positivas, su valoración y las palabras negativas y su valoración y después contrastarlas.

- **JavaPairReceiverInputDStream<String, String>**:

Esta clase es un receptor de Spark asociado a una fuente de datos. Esta fuente de datos es Kafka. El modelo de asociación es una suscripción a un tópico de Kafka.

```
// Parametros de configuración
SparkConf sparkConf = new
SparkConf().setAppName("AnalisisSentimientos").setMaster("local[
2]");
// creamos el contexto con lotes de mensajes leidos en 100
segundos
JavaStreamingContext jssc = new JavaStreamingContext(sparkConf,
new Duration(100000));
//creamos el consumidor de Kafka
JavaPairReceiverInputDStream<String, String> mensajes =
                KafkaUtils.createStream(jssc, String.class,
                            String.class,
                            StringDecoder.class,
StringDecoder.class,kafkaParams,topicMap,StorageLevel.MEMORY_AND
_DISK());
```

A partir de aquí obtenemos lotes de parejas de <tópico, tweet>. En el proceso siguiente descartamos el tópico y creamos un *JavaDStream<String> lineas*, para quedarnos solo con los tuits de los mensajes acumulados en el receptor.

```
//creamos un mapa de lineas sin el nombre del tópico
JavaDStream<String> lineas = mensajes.map(new
Function<Tuple2<String, String>, String>() {
      @Override
      public String call(Tuple2<String, String> tuple2) {
        return tuple2._2();
      }
  });
```

Una vez obtenidos los tuits hacemos una descomposición en palabras utilizando el carácter blanco como separador.

```
//creamos un flujo de palabras separadas por un blanco.
JavaDStream<String> palabras = lineas.flatMap(new
FlatMapFunction<String, String>() {
      @Override
      public Iterator<String> call(String x) {
        return Arrays.asList(x.split(" ")).iterator();
      }
    });
```

Ahora creamos tuplas <palabra, 1> y asociamos a cada palabra un valor de contador igual a 1.

```
//Creamos tuplas de palabra, y 1, quitamos tambien el # a las
palabras que emparejamos
JavaPairDStream<String, Integer> contadorPalabras =
palabras.mapToPair(
            new PairFunction<String, String, Integer>() {
              @Override
              public Tuple2<String, Integer> call(String s) {
                // quitamos el caracter # de los hashtag
                   if(s!=null && s.length()>0 && s.charAt(0)=='#')
                         s=s.substring(1);
                return new Tuple2<>(s, 1);
              }
            });
```

A continuación, agrupamos las palabras y sumamos sus contadores durante un tiempo de 100 segundos. Esto determinará los lotes de proceso.

```
//agrupamos palabras y contamos su aparición
JavaPairDStream<String, Integer> totalPalabras =
contadorPalabras.reduceByKeyAndWindow(
      new Function2<Integer, Integer, Integer>() {
        @Override
        public Integer call(Integer a, Integer b) {
          return a + b;
        }
      }, new Duration(100000));
```

Seguidamente leemos el fichero AFINN con las palabras que reflejan un estado sentimental.

```
// leemos el fichero de palabras asocias a estados sentimentales o
anímicos y creamos un RDD estático
String ficheroSentimientos= "……..AFINN.txt";
JavaPairRDD<String, Integer> palabrasSentimentales= jssc.sparkContext()
            .textFile(ficheroSentimientos)
            .mapToPair(new PairFunction<String, String, Integer>(){
              @Override
              public Tuple2<String, Integer> call(String linea) {
                String[] columnas = linea.split("\t");
                return new Tuple2<>(columnas[0],
Integer.parseInt(columnas[1]));
               }
            });
```

Ahora cruzamos ambos flujos, el flujo de palabras de tweet con el flujo estático de palabras sentimentales, para obtener un flujo formado por elementos que contienen una palabra, el número de veces que ha aparecido y su valoración sentimental que puede ser positiva o negativa.

```
// intersección entre las palabras sentimentales y las palabras de los
tweet
JavaPairDStream<String, Tuple2<Integer, Integer>> palabrasInterseccion =
            totalPalabras .transformToPair(new
Function<JavaPairRDD<String, Integer>,
              JavaPairRDD<String, Tuple2<Integer, Integer>>>() {
              @Override
              public JavaPairRDD<String, Tuple2<Integer, Integer>> call(
                  JavaPairRDD<String, Integer> contadorPalabras) {
                return palabrasSentimentales.join(contadorPalabras);
              }
});
```

Obtenido el flujo anterior ahora hay que transformarle para obtener un flujo de palabras con su valoración absoluta, es decir, el producto del número de veces que aparece por su valoración sentimental.

```
//transformación del flujo anterior para multiplicar el número de veces
que aparece la palabra por su valoración
JavaPairDStream<String, Integer> estadoSentimental =
palabrasInterseccion.mapToPair(
        new PairFunction<Tuple2<String, Tuple2<Integer, Integer>>, String,
Integer>() {
              @Override
              public Tuple2<String, Integer> call(Tuple2<String,
                Tuple2<Integer, Integer>> palabraSentimental) {
                Tuple2<Integer, Integer> contadorYValoracion =
palabraSentimental._2();
                return new Tuple2<>(palabraSentimental._1(),
                  contadorYValoracion._1() * contadorYValoracion._2());
              }
});
```

Ahora transformamos en flujo en otro en el que tenemos ordenadas las palabras en orden descendente (mayor valoración a menor). De esta forma tenemos ordenadas las palabras según su valoración sentimental.

```
//ordena segun el flag de sortbykey
JavaPairDStream<Integer, String> lasMasFelices =
estadoSentimental1.transformToPair(
            new Function<JavaPairRDD<Integer, String>,
JavaPairRDD<Integer, String>>() {
              @Override
              public JavaPairRDD<Integer, String> call(
                  JavaPairRDD<Integer, String> felicidadPalabra) {
                return felicidadPalabra.sortByKey(false); //descendiente-
-false (mayor a menor)
              }
            }
 );
```

A continuación, cogemos las tuplas de este flujo y las almacenamos en la base de datos de MySQL server para su posterior representación gráfica.

```
// Salva en MySQL todas las palabras empezando por las más felices
lasMasFelices.foreachRDD(new VoidFunction<JavaPairRDD<Integer,
String>>() {

            @Override
            public void call(JavaPairRDD<Integer, String>
estadoSentimental1) {
              List<Tuple2<Integer, String>> topList =
estadoSentimental1.take((int)estadoSentimental1.count());
              for (Tuple2<Integer, String> pair : topList) {
                   sql.escribir(pair._2(), pair._1());

              }

            }
});
```

Para operar con MySQL debemos abrir una sesión y escribir un DataSet formado por la tupla: palabra, valor y, a continuación, escribir cada tupla: (palabra,valor) en una tabla con un esquema representado por la clase Valoracion (sus atributos privados son palabra (String) y valor (int)) y los diferentes getters y setters, lo que se denomina bean.

```
    //abrimos una sesion con el servidor de base de datos
    SQLguardar sql = new SQLguardar();
```

```
public void escribir(String palabra, int valor) {
          Valoracion v=new Valoracion( );
          v.setPalabra(palabra);
          v.setValor(valor);

          Dataset<Valoracion> jdbcValor =
spark.createDataset(Collections.singletonList(v),
Encoders.bean(Valoracion.class));
          //jdbcValor.show();

          // Saving data to a JDBC source
          jdbcValor.write().mode(SaveMode.Append)
            .format("jdbc")
            .option("url", MYSQL_CONNECTION_URL)
            .option("dbtable", "sentimiento.analisis")
            .option("user", MYSQL_USERNAME)
            .option("password", MYSQL_PWD)
            .save();


      }
```

- **Maven**

  La configuración del proyecto utiliza Maven para el acceso a los paquetes necesarios para la generación del mismo.

```
<dependencies>
	<dependency>
		<groupId>org.apache.spark</groupId>
		<artifactId>spark-streaming_2.11</artifactId>
		<version>2.1.1</version>
	</dependency>

	<dependency>
		<groupId>org.apache.spark</groupId>
		<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>
		<version>2.1.1</version>
	</dependency>

	<dependency>
		<groupId>org.apache.spark</groupId>
		<artifactId>spark-sql_2.11</artifactId>
		<version>2.1.1</version>
	</dependency>

	<dependency>
		<groupId>mysql</groupId>
		<artifactId>mysql-connector-java</artifactId>
		<version>6.0.6</version>
	</dependency>
</dependencies>
```

4.4.4 VisualizacionGrafica

Esta aplicación se encarga de la representación gráfica del análisis realizado. Este análisis de sentimiento ha consistido en cuantificar las palabras positivas y negativas que aparecen en los tuits recibidos, lo que nos da una valoración global del estado de sentimiento positivo o negativo de las personas en una determinada ventana de tiempo.

El diseño de clases implementado y la arquitectura de la aplicación es el siguiente:

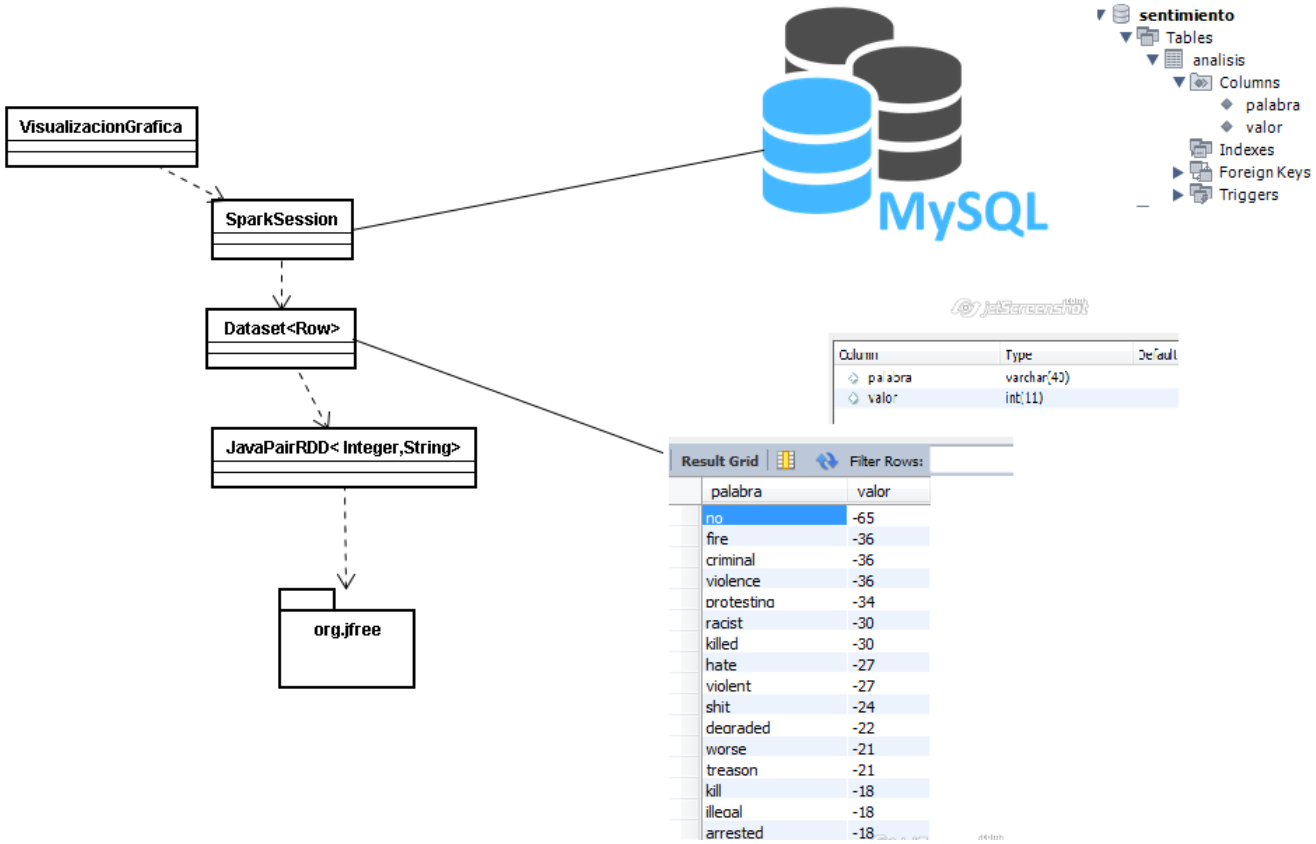

Figura 22: Diseño de clases de VisualizacionGrafica

En este diseño podemos ver las clases más importantes, así como la base de datos de almacenamiento de las palabras, su esquema y una tabla con las palabras y su valoración.

- **VisualizacionGrafica**

Esta es la clase principal desde la que se instancia un objeto del tipo SparkSession, y desde la que se accede al servidor de base de datos MySQL. Una vez obtenida la sesión leemos la tabla y la asociamos a un set de datos que tiene la misma estructura que la tabla definida en MySQL.

- **SparkSession**

Esta clase permite establecer una sesión con MySQL para leer la tabla como una fuente de datos. A través del método spark.read() se obtiene la tabla completa.

```
Properties connectionProperties = new Properties();
connectionProperties.put("user",MYSQL_USERNAME);
connectionProperties.put("password", MYSQL_PWD);
Dataset<Row> tabla = spark.read()
            .jdbc(MYSQL_CONNECTION_URL, "sentimiento.analisis",
connectionProperties);
```

Obtenida la tabla transformamos el set de datos para poder operar con palabra y su valor.

```
JavaPairRDD<String,Integer> resultRDD =
tabla.toJavaRDD().mapToPair(new
PairFunction<Row,String,Integer>() {

            @Override
            public Tuple2<String,Integer> call(Row row) {
                return new
Tuple2<String,Integer>(row.getString(0),row.getInt(1));
            }
        });
```

Como en la tabla hay lotes de procesamiento, hay que hacer una reducción utilizando las palabras de cada lote como clave con el objetivo de sumar las puntuaciones de cada palabra en los distintos lotes.

```
JavaPairRDD<String,Integer> reducirParejas =
resultRDD.reduceByKey( new Function2<Integer, Integer,
Integer>() {

      @Override
      public Integer call(Integer arg0, Integer arg1) throws
      Exception {return arg0+arg1;}
      }

);
```

Posteriormente convertimos el RDD en una lista de tuplas <Integer,String> ya que es el tipo de parámetro que requiere el objeto gráfico para hacer la representación gráfica.

```
//creamos una nueva lista añadiendo a la palabra su puntuación
List<Tuple2<Integer, String>> lista = new ArrayList<Tuple2<Integer,
String>>();
      for (Tuple2<Integer, String> t  :listaSentiNegOrd){

                  lista.add(new Tuple2<Integer,
String>(t._1,t._2+":"+ t._1));

             }
Graficas g = new Graficas();
g.MostrarGrafica(lista,NPALABRAS +" Palabras mas NEGATIVAS, de:
"+nPalabras);
```

Para las palabras positivas hacemos el mismo proceso que para las palabras negativas.

4.5 Puesta en marcha y visualización de resultados

El análisis representado se ha realizado durante 900 segundos, lo que equivalen a 15 minutos, el 18 de junio de 2017 a las 14:00.

El resultado obtenido se puede ver en la presentación de tres gráficas circulares:

Una, para la representación de las palabras con sentimiento positivo, otra con las palabras representando un sentimiento negativo y la última, la presentación global del estado de sentimiento.

También podemos ver que, de todos los tweets analizados, 999 palabras expresaban sentimientos, ya que están contenidas en la lista de palabras utilizada.

- **Gráfica de sentimientos negativos**

En esta representación gráfica circular podemos ver las 10 palabras más negativas y su puntuación:

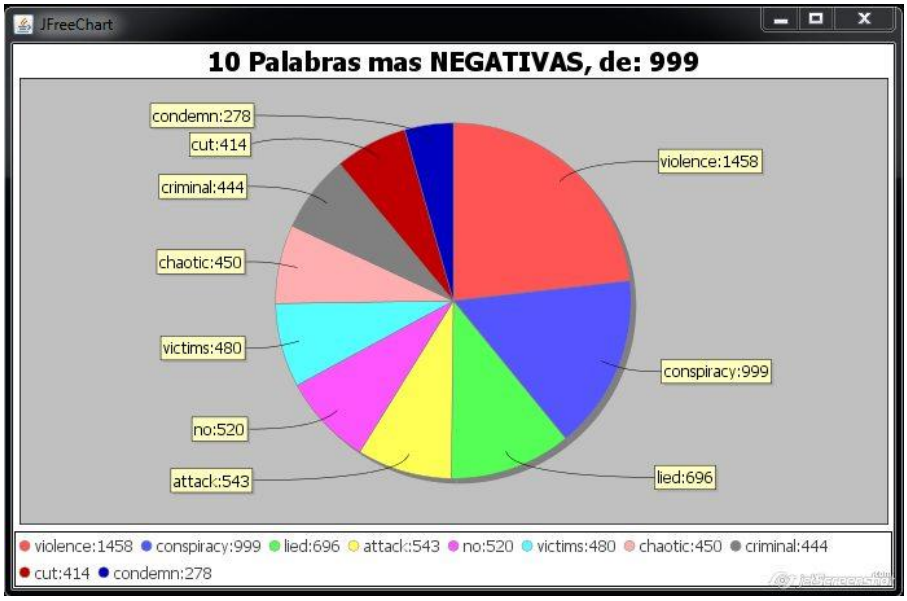

Figura 23: Gráfica 10 palabras más negativas

- **Gráfica de sentimientos positivos**

En esta representación gráfica circular podemos ver las 10 palabras más positivas y su puntuación:

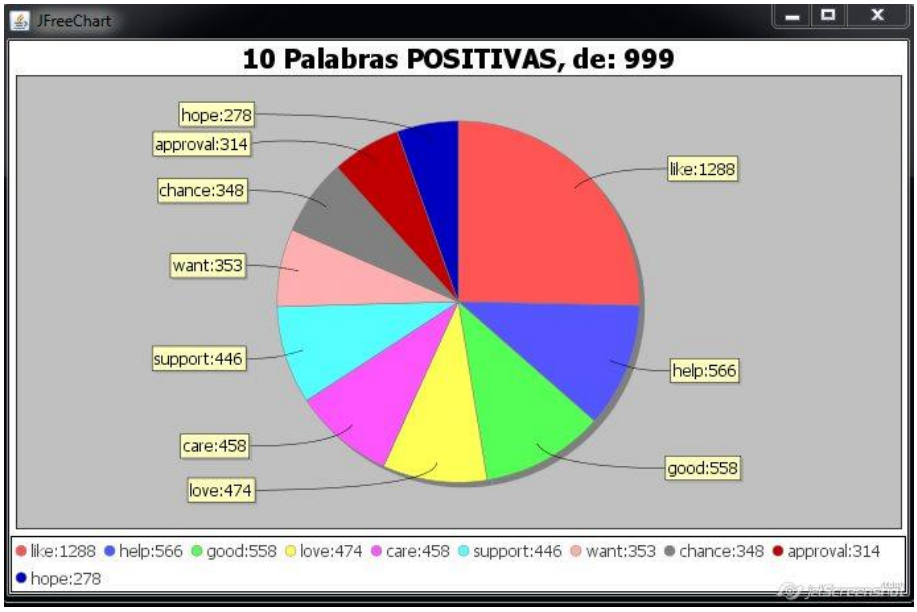

Figura 24: Gráfica 10 palabras más positivas

- **Gráfica de sentimiento general**

En esta representación gráfica circular vemos el balance global. Se obtiene sumando la valoración de todas las palabras con sentimiento positivo y todas las palabras con sentimiento negativo:

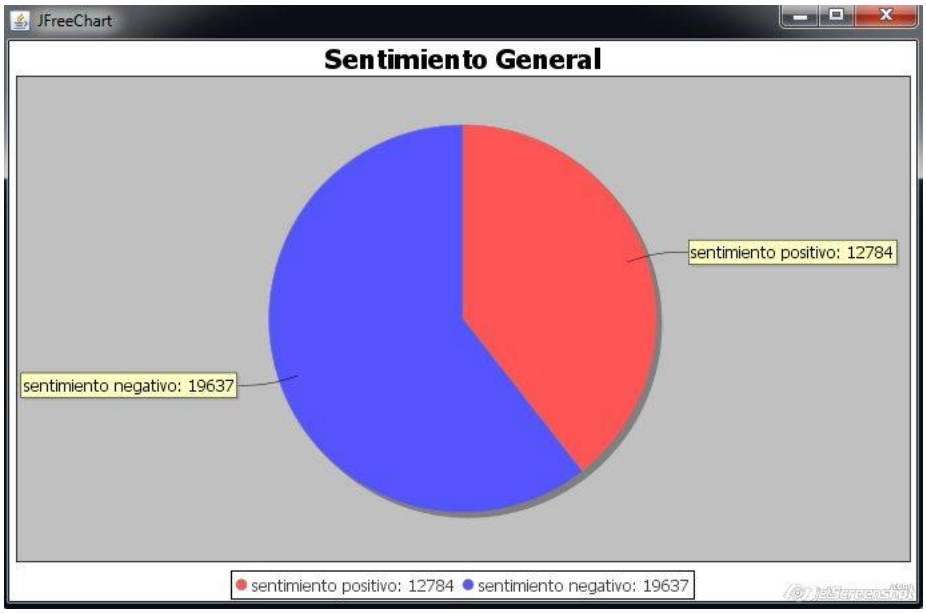

Figura 25: Gráfica sentimiento general

Podemos apreciar que, en la ventana de tiempo realizado, 15 minutos, el sentimiento negativo global es de un 60%, un 10 % superior al positivo. Luego, de este análisis podríamos inferir que el análisis de los tuits referidos a "Trump" tienen una valoración de sentimiento negativa.

CAPÍTULO 5

5 TRABAJOS RELACIONADOS

Existen numerosas herramientas online en las que se analiza la red social Twitter [40] como, por ejemplo:

- Sentiment140 [41], se trata de una herramienta gratuita que analiza los sentimientos de los tuits para así conocer las vibraciones que está transmitiendo una marca, un producto o una empresa en Twitter. No ha sido creada con las mismas herramientas que este proyecto, ya que se ha utilizado Amazon EC2 y distintas herramientas de Google, pero su función es similar.

Figura 26: Sentiment 140 [41]

- TweetPsych [42], consiste en una herramienta gratuita creada por Dan Zarrella, que analiza las características lingüísticas de los tuits de un determinado Twitter. Escanea los últimos 1000 tuits del usuario de Twitter que se introduzca. Además, presenta tuits con un contenido y pensamientos similares al de la cuenta introducida.

Figura 27: TweetPsych [42]

- Audiense [43], es una aplicación gratuita creada por dos españoles, desde donde se puede descubrir cuáles son los intereses de tus seguidores, cuál es la mejor hora para publicar un tuit, y algunas cosas más. Es decir, analiza Twitter para que así las empresas puedan sacar mucho más partido a sus campañas de marketing y sepan hacia qué público dirigirse.

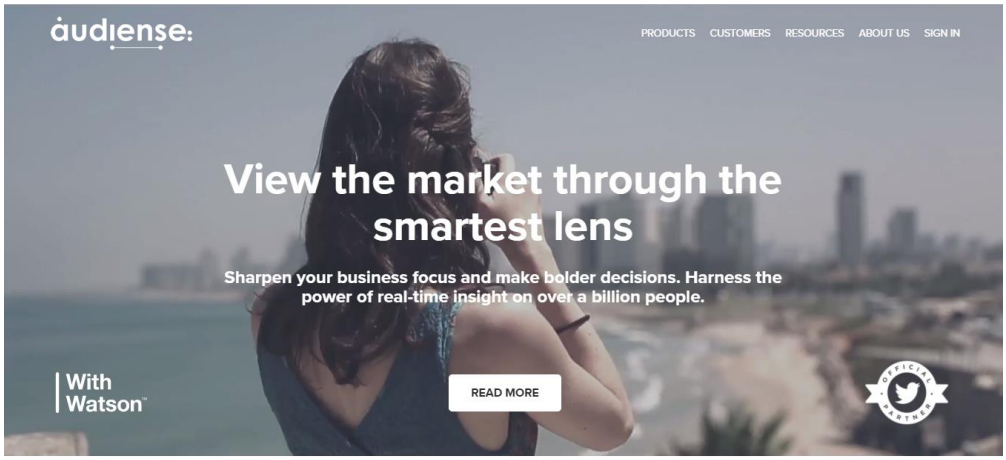

Figura 28: Audiense [43]

CAPÍTULO 6

6 IMPACTO SOCIO-ECONÓMICO

El impacto socio-económico de la aplicación desarrollada se puede apreciar en los siguientes aspectos:

El análisis de información utilizando tecnologías de Big Data tiene multitud de aplicaciones en diferentes áreas como ya lo hemos mencionado en el proyecto. Generar un conocimiento diferencial de los datos de las redes sociales y extraer información oculta, les puede permitir a las empresas obtener una ventaja sobre sus competidores.

Esto refleja un gran impacto social, en la forma en la que pueden actuar las compañías, y a su vez un impacto económico, puesto que analizar el estado anímico de un conjunto de personas es muy significativo ya que permitirá dirigir sus estrategias hacia un determinado objetivo, saber cómo satisfacer a sus clientes o mejorar sus campañas de marketing.

Este proyecto también puede suponer un impacto en la gestión política, puesto que conocer la opinión de los ciudadanos es a su vez muy beneficioso para realizar de forma adecuada las campañas electorales o, por ejemplo, para poder hacer un sondeo virtual de opinión.

Supone un gran ahorro de costes tanto para las empresas como para las entidades políticas, ya que es un análisis muy rápido y barato del que se puede extraer información muy productiva.

CAPÍTULO 7

7 MARCO REGULADOR

Mediante la tecnología Big Data, se pueden procesar grandes volúmenes de datos, con el fin de obtener información. El volumen, variedad y velocidad a la que se pueden procesar los datos, abre el camino a un amplio abanico de posibilidades sin precedentes. Pero el tratamiento de los datos de los usuarios genera la necesidad por parte de las empresas, de conocer y aplicar de forma adecuada la normativa legal de tratamiento de datos, para poder establecer una estrategia de cumplimiento de la misma.

En el marco regulador, hay que tener en cuenta aspectos como la seguridad, privacidad y conservación de los datos, así como preservar los derechos de los consumidores sobre sus datos.

Se considera un dato de carácter personal a toda información que posibilite la identificación de una forma directa o indirecta de una persona. Y por ello, el derecho fundamental de protección de datos consiste en otorgar al ciudadano la capacidad de disponer, controlar y decidir sobre sus datos personales.

La protección de datos personales tiene una importante dimensión internacional.

En el entorno europeo, las legislaciones nacionales están basadas en normas de la Unión Europea o del Consejo de Europa.

A escala global, el intercambio de datos ha crecido exponencialmente en los últimos años, lo que ha originado que se busquen distintos mecanismos de colaboración, al observarse la necesidad de acuerdo entre los distintos marcos legales de los diferentes países para posibilitar que el intercambio de datos garantice un nivel suficiente de protección.

En nuestro país, el organismo encargado de velar por el cumplimiento de la normativa de protección de datos es la Agencia Española de Protección de Datos (AEPD) [44].

El marco regulador a nivel nacional es la Ley Orgánica 15/1999, de 13 de diciembre, [45] de Protección de Datos de carácter personal. En dicha Ley Orgánica en su primer artículo 1, se indica el Objeto de la misma:

"La presente Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar. "

El Real Decreto 1720/2007, de 21 de diciembre, por el que se aprueba el Reglamento de desarrollo de la Ley Orgánica 15/1999, de 13 de diciembre, de protección de datos de carácter personal. Explica que:

"La actual Ley Orgánica 15/1999, de 13 de diciembre de Protección de datos de carácter personal adaptó nuestro ordenamiento a lo dispuesto por la Directiva 95/46/CE del Parlamento Europeo y del Consejo de 24 de octubre de 1995, relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos, derogando a su vez la hasta entonces vigente Ley Orgánica 5/1992, de 29 de octubre, de Regulación del tratamiento automatizado de datos de carácter personal."

Tal y como se explica en la página web de la Agencia Nacional de Protección de Datos [46]:

A nivel europeo, la principal normativa en materia de protección de datos es la Directiva 95/46 de la Unión Europea, como acabamos de comentar, esta normativa ha sido adoptada por nuestro ordenamiento nacional. Junto con el convenio nº108 del Consejo de Europa, de 28 de enero de 1981, para la protección de las personas respecto al tratamiento automatizado de datos de carácter personal, que es el único instrumento internacional vinculante sobre protección de datos.

Estas normativas, están siendo revisadas en la actualidad, para poder ajustarlas a los nuevos desarrollos tecnológicos y la globalización de los datos. Las modificaciones a la Directiva 95/46 se produjeron con la entrada en vigor del Tratado de Lisboa en el año 2009. Mientras que sobre el convenio nº108 existe una propuesta técnica que está todavía en estudio a nivel político.

CAPÍTULO 8

8 PLANIFICACIÓN Y PRESUPUESTO

8.1 Planificación

En este capítulo trataremos las tareas o actividades en las que se ha dividido el proyecto y su evolución temporal, empleando un diagrama de Gantt.

El proyecto se ha desarrollado en cinco meses aproximadamente. Ha sido supervisado por el tutor a través de reuniones periódicas y sus actividades han consistido en:

- Planteamiento y descripción del proyecto: Se plantea la idea general en la que se va a basar el proyecto.
- Estudio de Big Data: En esta fase se estudia los conceptos asociados al Big Data, las utilidades del mismo y se plantea el experimento que se va a realizar.
- Estudio de las tecnologías asociadas al Big Data: Se estudian y analizan más en profundidad las tecnologías que van a ser utilizadas en el desarrollo del proyecto.
- Definición del problema: Se plantea el problema en el que se va a centrar el proyecto: analizar sentimentalmente Twitter, la arquitectura del sistema y los requisitos que van a ser necesarios para la implementación del mismo.
- Implementación: Se instalan y configuran las herramientas que van a ser utilizadas. Se crea el código necesario para el funcionamiento del sistema.
- Pruebas y conclusiones: Se realizan pruebas del sistema para verificar el correcto funcionamiento del mismo y se realizan los cambios necesarios para corregir los errores que aparecen.
- Redacción de la memoria: Se redacta la memoria del proyecto.

En la tabla 1, vemos reflejadas las actividades comentadas anteriormente con fecha de inicio, duración y fecha de finalización de cada actividad:

ACTIVIDAD	FECHA INICIO	DURACIÓN	FECHA FINALIZACIÓN
Planteamiento y descripción del proyecto	01/02/2017	9	10/02/2017
Estudio de Big Data	11/02/2017	13	24/02/2017
Estudio de las tecnologías asociadas al Big Data	25/02/2017	20	16/03/2017
Definición del problema	17/03/2017	15	01/04/2017
Implementación	02/04/2017	43	15/05/2017
Pruebas y conclusiones	16/05/2017	25	10/06/2017
Redacción de la memoria	01/05/2017	44	14/06/2017

Tabla 1: Actividades realizadas

En la figura 29, se muestra el diagrama de Gantt para ver de forma gráfica la secuenciación de las tareas que se han llevado a cabo:

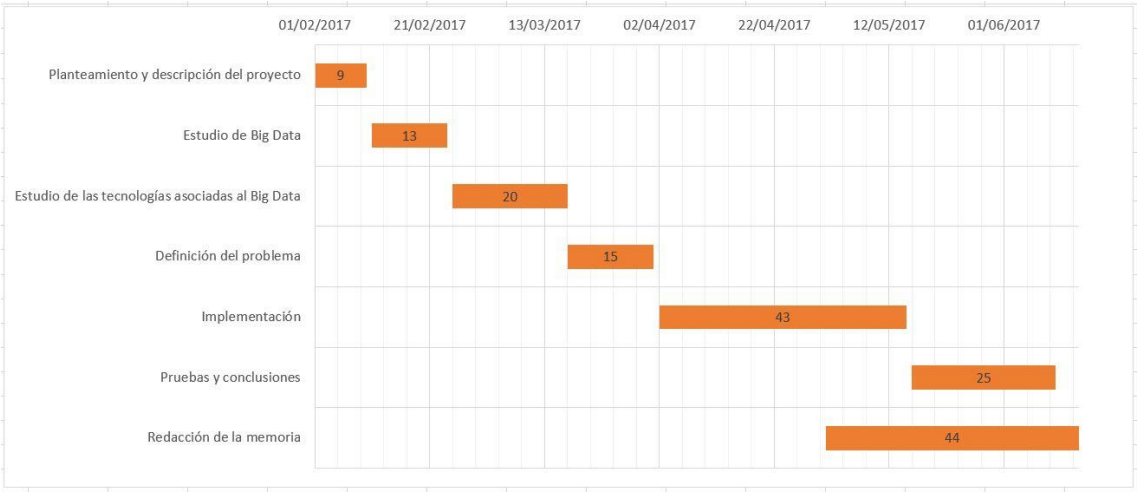

Figura 29: Diagrama de Gantt

8.2 Presupuesto

En este apartado se detallará el presupuesto necesario para llevar a cabo el proyecto.

8.2.1 Recursos utilizados

En primer lugar, trataremos los recursos utilizados en el proyecto con una descripción de su función. Los recursos se dividen en dos tipos:

- Recursos humanos: Se trata de las personas encargadas de la organización y realización de las actividades.
    - Tutor del proyecto: Con categoría de ingeniero senior, ha sido el encargado de la organización y dirección del proyecto.
    - Autor del proyecto: Con categoría de ingeniero junior, persona responsable de llevar a cabo el desarrollo del proyecto, desde el estudio y análisis de las tecnologías hasta la implementación de la aplicación.
- Recursos materiales: son los elementos o componentes de software y hardware que han sido utilizados
    - Ordenador portátil DELL: Latitude 3340
    - Microsoft Office 2016
    - Eclipse Java Neon
    - Kafka: versión 2.12
    - Spark: versión 2.10
    - MySQL: versión 5.7.18.1
    - API de Twitter

8.2.2 Costes

Una vez definidos los recursos que han sido utilizados, en las siguientes tablas se muestran los costes asociados a dichos recursos.

En la tabla 2, se pueden ver los costes materiales:

COSTES MATERIALES	
MATERIAL	COSTE
PC portátil DELL 8gb RAM	600 €
Microsoft Office 2016	69 €
Eclipse	0 €
Spark	0 €
Kafka	0 €
MySQL	0 €
Twitter	0 €
TOTAL	669€

Tabla 2: Costes materiales

En la tabla 3, se muestran los costes del personal que ha participado el proyecto, teniendo en cuenta el coste por horas y las horas dedicadas:

COSTE DEL PERSONAL				
Cargo	Número	Coste por horas	Horas dedicadas	Coste total
Ingeniero senior	1	25 €	20	500 €
Ingeniero junior	1	12 €	420	5.040 €
			TOTAL	5540€

Tabla 3: Coste del personal

Por último, podemos ver en la tabla 4 el coste total estimado para el proyecto:

COSTES TOTALES	
Costes materiales	669 €
Costes de personal	5.540 €
TOTAL	6209€

Tabla 4: Costes totales

CAPÍTULO 9

9 CONCLUSIONES Y TRABAJOS FUTUROS

En la fase de desarrollo se han realizado múltiples consultas a las páginas oficiales de Apache sobre Kafka y Spark y a otras páginas para la consulta de ayuda sobre los múltiples problemas que iban apareciendo: como incompatibilidad entre versiones de los diferentes sistemas, configuración de los sistemas, diferentes modos de resolver un determinado problema, etc. Hay que decir que se ha recibido más ayuda de webs donde los usuarios exponían sus problemas que de las páginas oficiales de las aplicaciones. La documentación y ejemplos de las páginas oficiales están pensadas, a mi modo de ver, para profesionales que con poco ayuda son capaces de resolver el problema, pero en mi caso necesitaba una información mucho menos técnica y más básica que solo encontraba en páginas de desarrolladores.

En cuanto al análisis del estado de sentimiento de la gente hay que contextualizarlo, en mi caso he elegido el nombre del presidente americano Trump. El tiempo de análisis ha sido en un intervalo de 15 minutos. Con más tiempo la precisión de la valoración sentimental podría haber sido mejor al tener mayor número de tweets. Si quisiéramos reducir el campo de análisis podríamos añadir no sólo "Trump" si no otra palabra clave, como podría ser "CIA".

El proyecto se ha desarrollado con un solo ordenador, no en modo clúster, es decir, tener varias máquinas trabajando en paralelo. Trabajando en paralelo con varias máquinas el número de tweets que seríamos capaces de procesar sería mayor, por lo que el resultado obtenido también resultaría más preciso.

Las aplicaciones Spark y Kafka están pensadas para trabajar en modo clúster, es decir, modo distribuido habiendo varias máquinas trabajando en paralelo. El mejorar el rendimiento del sistema montando un clúster es un posible trabajo futuro.

En la aplicación implementada se realiza el análisis y terminado el análisis se visualizan los resultados. Por lo que otra de las mejoras que pueden realizarse es la presentación visual de los resultados en tiempo real, es decir, según se depositan en la base de datos los resultados ir actualizando las gráficas.

Y como conclusión, después de haber realizado el desarrollo y la implementación de la aplicación se puede decir que se ha cumplido con los objetivos propuestos y todas las complicaciones han sido solucionadas de la mejor manera posible.

CAPÍTULO 10

10 BIBLIOGRAFÍA

[1] M. Á. Lucas. [En línea]. Available: https://mibloguel.com/big-data-significado-y-su-utilidad-en-la-sociedad/. [Último acceso: 10 06 2017].

[2] J. C. López, «eleconomista,» 27 02 2014. [En línea]. Available: http://www.eleconomista.es/tecnologia/noticias/5578707/02/14/La-moda-del-Big-Data-En-que-consiste-en-realidad.html. [Último acceso: 10 06 2017].

[3] Mikel, «Cronología del Big Data,» [En línea]. Available: http://www.mikelnino.com/2015/09/cronologia-big-data.html. [Último acceso: 02 06 2017].

[4] B. Marr, «mediapostgroup,» 25 05 2016. [En línea]. Available: http://www.mediapostgroup.es/blog/las-5-vs-del-big-data/. [Último acceso: 10 06 2017].

[5] «Big Data,» 21 04 2016. [En línea]. Available: http://bigdata.black/featured/what-is-big-data/. [Último acceso: 02 06 2017].

[6] J. Faber, «Fuentes de datos,» 27 06 2016. [En línea]. Available: https://johnfaberblog.wordpress.com/2016/07/27/fuentes-de-datos/. [Último acceso: 10 06 2017].

[7] «ciospain,» 03 04 2013. [En línea]. Available: http://www.ciospain.es/big-data/cuales-son-las-principales-fuentes-del-big-data. [Último acceso: 10 06 2017].

[8] BBVAInnovationCenter, «bbva,» 30 01 2015. [En línea]. Available: http://www.centrodeinnovacionbbva.com/noticias/ejemplos-reales-del-uso-de-big-data. [Último acceso: 10 06 2017].

[9] «Big Data Landscape,» 25 02 2016. [En línea]. Available: http://todobi.blogspot.com.es/2016/02/big-data-landscape-2016.html. [Último acceso: 02 06 2017].

[10] P. G. Bejerano, «Hadoop open source,» 09 08 2013. [En línea]. Available: http://blogthinkbig.com/hadoop-open-source-big-data/. [Último acceso: 10 06 2017].

[11] Hortonworks, «Beneficios Hadoop,» [En línea]. Available: https://es.hortonworks.com/apache/hadoop/. [Último acceso: 10 06 2017].

[12] «Arquitectura Hadoop,» 06 04 2015. [En línea]. Available: https://elentornodehadoop.wordpress.com/tag/arquitectura-hadoop/. [Último acceso: 10 06 2017].

[13] wikipedia, «MapReduce,» [En línea]. Available: https://es.wikipedia.org/wiki/MapReduce. [Último acceso: 10 06 2017].

[14] «Ejemplo Map Reduce,» 08 04 2015. [En línea]. Available: https://pmonterom.wordpress.com/. [Último acceso: 02 06 2017].

[15] HadoopApacheOrg, «Hdfs,» [En línea]. Available: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html. [Último acceso: 10 06 2017].

[16] BBVAopen4u, «Apache Spark las ventajas de usar al nuevo rey de big data,» 25 09 2015. [En línea]. Available: https://bbvaopen4u.com/es/actualidad/apache-spark-las-ventajas-de-usar-al-nuevo-rey-de-big-data. [Último acceso: 10 06 2017].

[17] SparkApacheOrg, «RDD,» [En línea]. Available: https://spark.apache.org/docs/1.6.2/api/java/org/apache/spark/rdd/RDD.html. [Último acceso: 10 06 2017].

[18] «Apache Spark RDD,» [En línea]. Available: https://www.tutorialspoint.com/apache_spark/apache_spark_rdd.htm. [Último acceso: 10 06 2017].

[19] SparkApacheOrg, «SparkContext,» [En línea]. Available: https://spark.apache.org/docs/0.6.2/api/core/spark/SparkContext.html. [Último acceso: 10 06 2017].

[20] J. Scott, «Quick Guide Spark Streaming,» 06 10 2015. [En línea]. Available: https://mapr.com/blog/quick-guide-spark-streaming/. [Último acceso: 10 06 2017].

[21] SparkApacheOrg, «StreamingContext,» [En línea]. Available: http://spark.apache.org/docs/0.7.3/api/streaming/spark/streaming/StreamingContext.html. [Último acceso: 10 06 2017].

[22] BBVAOpen4u, «Tres alternativas solidas de big data en tiempo real spark storm y datatorrent rts,» 07 08 2015. [En línea]. Available: https://bbvaopen4u.com/es/actualidad/tres-alternativas-solidas-de-big-data-en-tiempo-real-spark-storm-y-datatorrent-rts. [Último acceso: 10 06 2017].

[23] E. G. Rubio, «BlogGfi,» 27 06 2014. [En línea]. Available: http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/. [Último acceso: 10 06 2017].

[24] «jansipke,» 30 01 2013. [En línea]. Available: https://jansipke.nl/installing-a-storm-cluster-on-centos-hosts/.

[25] T. Engineering. [En línea]. Available: http://www.treselle.com/blog/twitter-analysis-with-apache-storm/. [Último acceso: 02 06 2017].

[26] KafkaApacheOrg. [En línea]. Available: https://kafka.apache.org/. [Último acceso: 10 06 2017].

[27] TutorialsPoint, «Apache kafka quick guide,» [En línea]. Available: https://www.tutorialspoint.com/apache_kafka/apache_kafka_quick_guide.htm. [Último acceso: 10 06 2017].

[28] I. Herrero, 23 08 2016. [En línea]. Available: http://www.biblogtecarios.es/inmaherrero/el-analisis-de-sentimiento-de-texto-en-las-redes-sociales/. [Último acceso: 10 06 2017].

[29] F. B. Márquez, «Acquiring and Exploiting Lexical Knowledge for Twitter Sentiment Analysis,» 2017. [En línea]. Available: http://www.cs.waikato.ac.nz/~fjb11/thesis.pdf. [Último acceso: 10 06 2017].

[30] «AFINN,» [En línea]. Available: http://neuro.imm.dtu.dk/wiki/AFINN. [Último acceso: 10 06 2017].

[31] «wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Twitter. [Último acceso: 10 06 2017].

[32] K. Smith, «Estadisticas Twitter 2016,» 07 06 2016. [En línea]. Available: https://www.brandwatch.com/es/2016/06/44-estadisticas-twitter-2016/. [Último acceso: 10 06 2017].

[33] DevTwitter. [En línea]. Available: https://dev.twitter.com/oauth/overview/application-owner-access-tokens. [Último acceso: 10 06 2017].

[34] ManageEngine, «Twitter Settings,» [En línea]. Available: https://www.manageengine.com/products/support-center/help/adminguide/configurations/helpdesk/twitter-settings.html. [Último acceso: 02 06 2017].

[35] SparkApacheOrg, «streaming programming guide,» [En línea]. Available: http://spark.apache.org/docs/latest/streaming-programming-guide.html. [Último acceso: 10 06 2017].

[36] «MySQL,» [En línea]. Available: https://www.mysql.com/. [Último acceso: 10 06 2017].

[37] «wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/MySQL. [Último acceso: 10 06 2017].

[38] Eclipse. [En línea]. Available: https://eclipse.org/.

[39] Twitter4j. [En línea]. Available: http://twitter4j.org/en/index.html. [Último acceso: 02 06 2017].

[40] «Analisis textual aplicaciones web social,» 05 12 2012. [En línea]. Available: https://wwwhatsnew.com/2012/12/05/analisis-textual-algunas-aplicaciones-web-social/. [Último acceso: 02 06 2017].

[41] sentiment140. [En línea]. Available: http://www.sentiment140.com/. [Último acceso: 10 06 2017].

[42] tweetpsych. [En línea]. Available: http://tweetpsych.com/. [Último acceso: 10 06 2017].

[43] audiense. [En línea]. Available: https://es.audiense.com/. [Último acceso: 10 06 2017].

[44] Agpd. [En línea]. Available: http://www.agpd.es/portalwebAGPD/canaldocumentacion/legislacion/estatal/index-ides-idphp.php. [Último acceso: 10 06 2017].

[45] «BOE,» [En línea]. Available: https://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750. [Último acceso: 10 Junio 2017].

[46] Agpd. [En línea]. Available: http://www.agpd.es/portalwebAGPD/internacional/Europa/index-ides-idphp.php. [Último acceso: 10 06 2017].

ANEXOS

Anexo A: Extended Abstract

1. Introduction

Context and motivation

Information has always been a competitive advantage for countries, companies and organizations. Owning this advantage is extremely important in order to achieve your goals or objectives and to be above your competitors.
The information we possess is growing exponentially in recent years, in such a way, that it has been a great problem to store it, and therefore to be able to analyze and interpret it.

In order to solve the problem of how to store such data, the technologies have evolved so that new storage devices with larger capacity and smaller size have appeared. And with respect to the analysis and interpretation of such information, technologies have been forced to increase the speed and capacity of the process to be able to do so.

That is why it appears what is known as Big Data to be able to analyze and obtain valuable information of this large amount of data that surrounds us today. [1]

In this project a large amount of data will be extracted from a social network in real time, making use of Big Data technologies, and the extracted data will be analyzed sentimentally. It comes from the idea that people make use of social networks continuously to express their opinion and their feelings towards any subject. So, extracting this information and analyzing it can be very useful, for example, for companies, as they can analyze how their customers are reacting to a particular product; and it can also have great utility in politics because it allows political entities to know how many people they have against or in favor.

Objectives

This project has pursued the following objectives:

1. Conduct a study of Big Data and existing technologies and architectures for large amount of data analysis.

2. Sentiment analysis of the social network Twitter, for this it is necessary to design a Big Data system to extract the tweets in streaming. Then the extracted data must be processed to obtain the feeling of those messages. And finally store and visualize the results.

3. Analyze the socio-economic impact of the project.

4. Analyze the regulatory framework associated with Big Data.

5. Conclusions from this analysis and propose improvements.

Document structure

1. Introduction: This chapter describes the context in which the project is developed and the motivation that leads to the project. The main objectives are established and the structuring of the memory is exposed.
2. State of the Art: This chapter analyzes Big Data, the main concepts around it, data sources and some applications in which its role is very useful.
3. Big Data technologies: Chapter three presents the main technologies that are used for the massive analysis of static data and data in real time. Some of them are necessary for the elaboration of the realized case.
4. Developed application: This chapter explains what is the meaning of sentiment analysis on Twitter, the system architecture, the implementation that has been carried out and the results obtained.
5. Related jobs: This chapter presents other works that are related with this project.
6. Socio-economic impact: This chapter analyzes the social and economic impact of the project.
7. Regulatory Framework: Chapter 7 sets out the regulatory framework surrounding Big Data.
8. Planning and budget: This chapter describes the stages of project development and the budget needed to carry it out.

9. Conclusions and future lines of work: This chapter analyses the achievement of the objectives and describes future improvements.
10. Bibliography: The sources of information that have been used in the preparation of the project are listed.
11. Annex A: It is made a summary of the report in english.

2. State of the art in Big Data

With the constant growth of data on the internet, a new term, called Big Data, emerges. Big Data is the term that describes the process of collecting a large volume of data, which can be both structured and unstructured, and obtain knowledge of this data.

This large amount of data stored grows exponentially day by day. [2]
To characterize the system and explain the advantages of Big Data five rules are used, the 5 V's [4]:

• Volume: Since the volume of data with which it works is enormous.

• Speed: The data flows at an unprecedented speed, so the speed at which they are analyzed is extremely important.

• Variety: The data have many types of structures and formats.

• Veracity: It is necessary to ensure the veracity of the information extracted.

• Value: It is the most relevant aspect of Big Data, as the volume and complexity of the data increases, its marginal value decreases considerably, due to its difficulty of exploitation.

Data sources

The different data sources [6] are very varied, so this causes that there is no homogeneity in the way of extracting them and later treating them. There is a wide range of sources from which we can extract information: Open Data, Social Networks, Internet of Things (IOT), etc.

Utilities of Big Data

Let's see some examples [8] in which the use of Big Data has brought great advantages:

• Macy's: it is one of the most important stores in the United States. Using Big Data technology Macy's can know exactly the impact of their news or the satisfaction of their clients.

• NBA teams: use data analytics to prepare their strategy and make the best decisions in each match.

• Obama's re-election: with their analytics, Obama team could optimize communication and did not waste resources on voters who were against him.

3. Technologies for Big Data analysis

Big Data systems

Apache Hadoop

Apache Hadoop is an open source system that facilitates storage, data processing and allows large amounts of information to be analyzed quickly and with high fault tolerance.

The modules of Hadoop are:

- Hadoop Common: Contains the libraries and file systems required to run Hadoop.
- Hadoop Yarn: It is a platform for managing and planning the resources of the cluster.
- Hadoop MapReduce: This is the programming model used by Hadoop to process large amounts of data in parallel.
  It is based on the realization of two operations, Map and Reduce to the data stored in the cluster [13]. It is made up of [12]: a JobTracker per cluster, distributes the works of MapReduce to the rest of TaskTrackers to make them. And by TaskTracker, there is one per

cluster node, which performs MapReduce operations on the data set it receives.

- Hadoop HDFS: It is Hadoop's distributed file system. It has great scalability, allows to distribute the information in clusters and is written in Java. It is a set of distributed nodes in which information and its replicas are stored. The data is divided into different blocks and stored in different machines or nodes to guarantee a good tolerance to failures and greater reliability of the system.
  Nodes are grouped into clusters, and on each cluster we find a NameNode, which takes care of carrying an access control and contains a record of the distribution of all the data stored in the system, and one or more DataNodes, that contain the information and are responsible for executing the client's read and write requests.

Apache Spark

Spark [16] is an open source framework for analyzing large volumes of data, built on Hadoop. It was designed to meet three priorities: speed, ease of use and advanced analytical capability.

Apache Spark is the new star of Big Data. The platform is written in scala, but can be operated in three other programming languages: Java, R and Python.

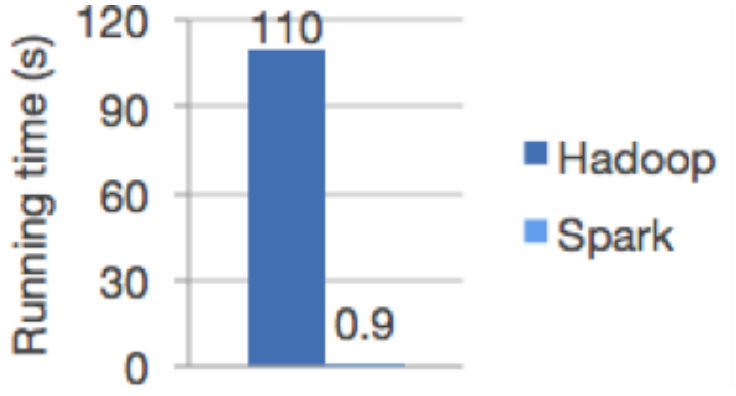

Figure 1: Running times of Hadoop and Spark

The key to its performance is that Spark provides the advantage that data can be loaded into a cluster memory and be queried and processed repeatedly.

Spark incorporates tools of great utility, such as:

- MLib library: to implement automatic learning solutions.
- GraphX: API for graphing.
- Spark Streaming: for constant input data processing in real time.
- Spark SQL: for the exploitation of the data by means of SQL language.

Resilient Distributed Datasets (RDD) [17] is the fundamental data unit that the Spark processing engine consumes. [18] Represents an immutable collection of objects, fault tolerant that can be operated in parallel.

Streaming systems

Apache Spark Streaming

Apache Spark Streaming [20] is an Apache Spark tool used for constant stream data processing.

Internally, Spark Streaming receives a stream of data and splits it into batches of data, which are then processed by Spark, which in turn returns batches of data already processed. This operation can be seen in figure 2:

Figure 2: Spark Streaming

It receives a continuous stream of data and converts it into a discrete stream, called DStream, which consists of data packets. A DStream is represented as a sequence of RDD (Resilient Distributed Data) objects.

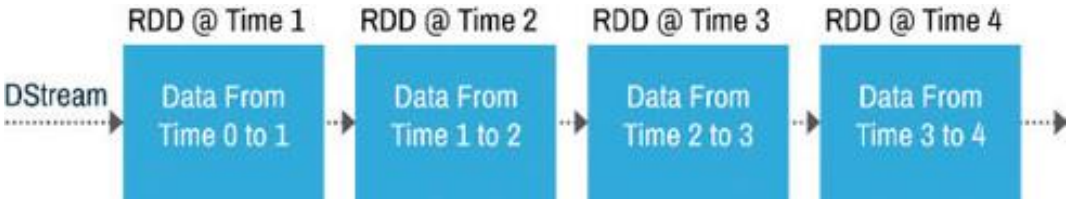

Figure 3: DStream

Micro-batches [20] on which Spark Streaming operates have a time interval between them. The main consequence is that the data can be received and processed in a different order from what actually happened. That is why platforms such as Apache Kafka are used, among others, to ensure that the data arrives in the exact order.

Apache Storm

Apache Storm [22] is a distributed, open-source real-time computing system. Storm process data streams reliably and without limits. Storm is simple and can be used with several programming languages like Java, Python, Scala, C #, among others.

Storm transforms data and analyzes it into a continuous process of constant input of information. Storm architecture is:

- Master or Nimbus: It is responsible for assigning and monitoring the different tasks on the cluster nodes.
- Slave or Supervisor: Collect and perform assigned tasks.

It is formed by Topologies, which are complex trees where compute is realized. Topologies are composed by spouts, in charge of collecting incoming data flow, and bolts, contains the operations to perform the calculations.

Complementary system

Apache Kafka

Apache Kafka [27] is a robust queue and a distributed messaging system with which it can handle a large volume of data and allows to pass messages. Kafka integrates nicely with Apache Spark and Apache Storm for real-time data analysis.

Kafka supports the delivery of low latency messages and guarantees fault tolerance in the presence of possible machine failures. It has the ability to handle a large number of diverse consumers and perform 2 million scripts per second.

It is necessary to build it on the ZooKeeper synchronization service.

4. Developed application

The application consists of performing a sentiment analysis of a social network. The selected social network has been Twitter. This analysis requires the knowledge and use of Big Data because it is necessary to process a large amount of information in streaming.

To carry it out, it is necessary to install, configure and start up a system formed by a streaming source (Twitter), a system to guarantee the integrity of the information received (Kafka), a streaming information analysis system (Spark Streaming) and a database for storing the results of the analysis (MySQL).

Sentiment analysis of Twitter

Sentiment analysis [28] of messages in social networks is the process through which the emotional tone behind a text can be determined, classifying the polarity expressed as positive, negative or neutral.

Given the great amount of emotions, opinions and evaluations seen daily in social networks, it is very useful to extract this information and to find out if the predominant feeling is positive or negative. The primary objective is to extract the terms that express a feeling to know the opinion and the sentimental state of the users, in order to be able to determine an impact or to be able to anticipate a certain reaction.

In this project, the tools of Big data have been used to extract large amounts of messages from the social network Twitter and to analyze them, through a list of words that express emotions with which those messages have been compared and extracted conclusions.

System architecture

Figure 4: Architecture of the application

Design and implementation of the system

The implemented system has been developed on a Windows 10 machine, using Eclipse [38] and the Java programming language version 1.8.f.

The system consists of three applications: KafkaTweetProducer, SparkTweetConsumer and Graphic Visualization.

Each of these console applications run separately, but KafkaTweetProducer and SparkTweetConsumer must run simultaneously as each one plays a complementary role. KafkaTweetProducer accesses the tweets and leaves them in Kafka available to Spark, represented by the SparkTweetConsumer application, which reads them, processes them and save the results in MySQL database to be accessed and presented graphically by the Graphic Visualization application.

The analysis was performed using the word "Trump" as a filter. This filter has been chosen because it is a current famous person.

Results visualization

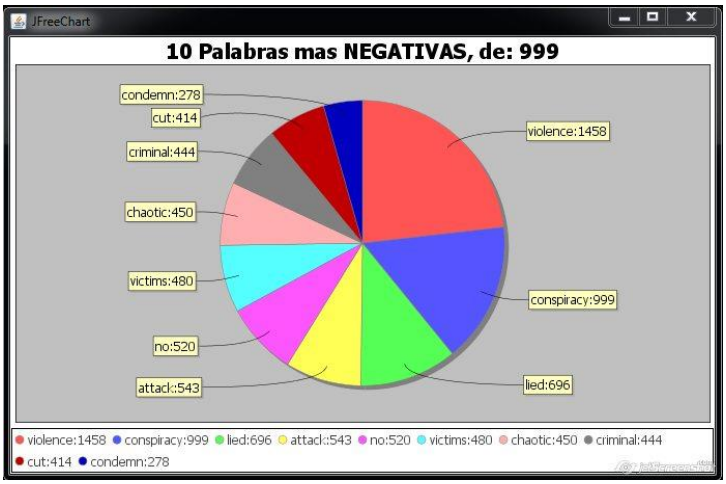

Figure 5: 10 most negative words

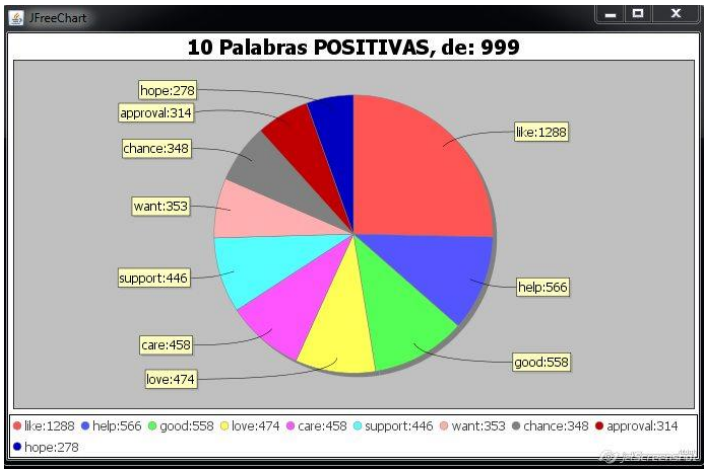

Figure 6: 10 most positive words

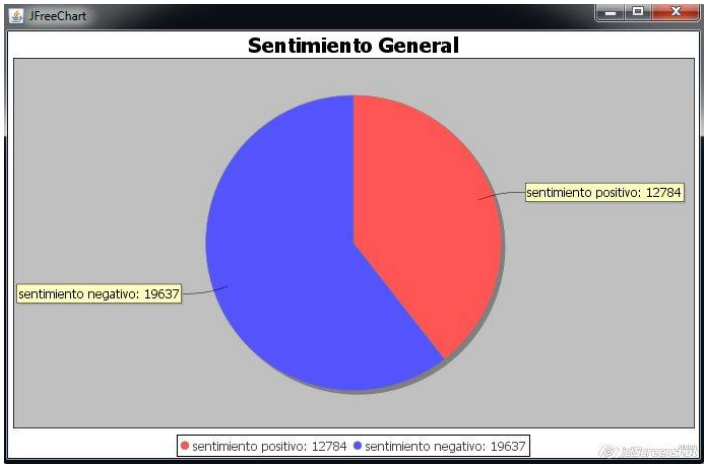

Figure 7: General sentiment

We can see that in the time window, 15 minutes, the overall negative feeling is 60%, 10% higher than the positive one. Then, from this analysis we could infer that the analysis of tweets referred to "Trump" have a negative feeling assessment.

5. Social and economic impact

Generating a differential knowledge of social networking data and extracting hidden information may allow companies to gain an advantage over their competitors.

This reflects a great social impact, in the way in which companies can act, and also, an economic impact, since to analyze the mental state of a group of people is very significant since it will allow directing its strategies towards a certain objective,

and they can know how to satisfy their clients or how to improve their marketing campaigns.

It is a great cost savings for both companies and political entities, since it is a very fast and cheap analysis from which they can extract very interesting and useful information.

6. Regulatory framework

Using Big Data technology, large volumes of data can be processed in order to obtain information. The volume, variety and speed at which data can be processed opens the way to a wide range of unprecedented possibilities. But the treatment of user data generates the need for companies to know and apply properly the legal regulations of data processing, to be able to establish a compliance strategy.

In the regulatory framework, we must consider aspects such as security, privacy and data preservation, as well as preserve the rights of consumers about their data.

7. Planning and budget

The project has been developed in approximately five months. It has been supervised by the tutor through periodic meetings and his activities have been:

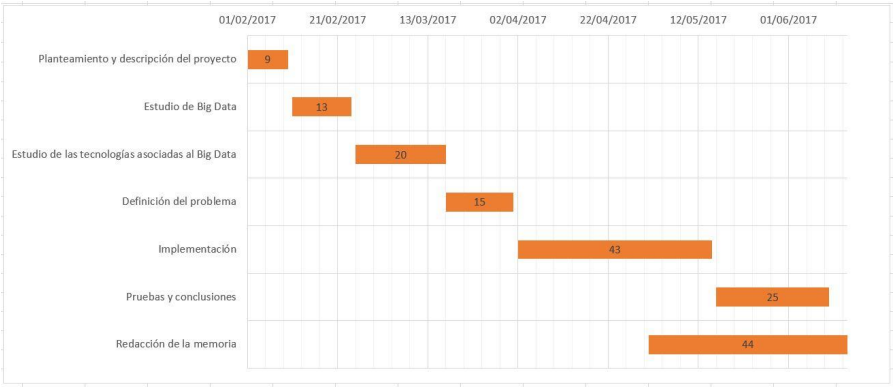

Figure 8: Gantt DIagram

The budget necessary to carry out the project considering the resources that have been used is:

COSTES TOTALES		
Costes materiales		669 €
Costes de personal		5.540 €
	TOTAL	6209€

Figure 9: Total cost

8. Conclusions and future lines of work

In the development phase, there have been multiple queries to the official Apache pages about Kafka and Spark and to other pages for the query of help on the multiple problems that were appearing: as incompatibility between versions of the different systems, configuration of the systems, different ways of solving a particular problem, etc.

The analysis time has been in an interval of 15 minutes. With more time the precision of the sentimental evaluation could have been better, having more number of tweets.

The project was developed with a single computer, not in cluster mode Working in parallel with several machines the number of tweets we would be able to process would be greater, so the result obtained would also be more accurate. Improving system performance by assembling a cluster is a possible future work.

In the implemented application, the analysis is performed and when it is finished, the results are displayed. So, another improvement that can be done is the visual presentation of the results in real time.

As conclusion, after having realized the development and the implementation of the application, it can be said that the proposed objectives have been fulfilled and all the complications have been solved in the best possible way.